L Number	Hits	Search Text	DB	Time stamp
1	6573	dma with transfer$1	USPAT; US-PGPUB	2003/05/13 19:09
4	80	subset$1 same intermittent	USPAT; US-PGPUB	2003/05/13 19:09
7	4	(subset$1 same intermittent) and (dma with transfer$1)	USPAT; US-PGPUB	2003/05/13 19:07
10	3772	dma with transfer$1	EPO; JPO; DERWENT; IBM_TDB	2003/05/13 19:09
15	6	subset$1 same intermittent	EPO; JPO; DERWENT; IBM_TDB	2003/05/13 19:09
20	0	(dma with transfer$1) and (subset$1 same intermittent)	EPO; JPO; DERWENT; IBM_TDB	2003/05/13 19:09
25	29	(subset$1 or partial$3) same (dma with transfer$1)	EPO; JPO; DERWENT; IBM_TDB	2003/05/13 19:15
30	144	(subset$1 or partial$3) same (dma with transfer$1)	USPAT; US-PGPUB	2003/05/13 19:16
33	11	30.ti,clm,ab.	USPAT; US-PGPUB	2003/05/13 19:21
36	70	(cycle adj steal) same (dma with transfer$1)	USPAT; US-PGPUB	2003/05/13 19:22
39	34	(transfer adj request$3) and ((cycle adj steal) same (dma with transfer$1))	USPAT; US-PGPUB	2003/05/13 19:34
42	1734	interrupt same (dma with transfer$1)	USPAT; US-PGPUB	2003/05/13 19:34
45	13	(burst adj mode) same (interrupt same (dma with transfer$1))	USPAT; US-PGPUB	2003/05/13 19:35

US-PAT-NO: 6115767

DOCUMENT-IDENTIFIER: US 6115767 A

TITLE: Apparatus and method of partially transferring data through bus and bus master control device

DATE-ISSUED: September 5, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Hashimoto; Yuichi	Osaka	N/A	N/A	JP
Kakiage; Touru	Neyagawa	N/A	N/A	JP
Suzuki; Masato	Toyonaka	N/A	N/A	JP
Kasuga; Yoshiaki	Shiga-gun	N/A	N/A	JP
Yasui; Jyunichi	Takatsuki	N/A	N/A	JP

US-CL-CURRENT: 710/107, 710/25 , 710/28 , 710/32 , 710/34 , 710/35 , 710/45

ABSTRACT:

A method of transferring data through a bus includes the steps of: occupying the bus by a first device serving as a bus master; transferring a first predetermined number of data items of all data items to be transferred while the first device is occupying the bus; determining if the first predetermined number of data items have been transferred; determining if the first device should release the bus based on whether or not there is a request from a second device after it is determined that the first predetermined number of data items have been transferred; and releasing the bus by the first device when it is determined that the first device should release the bus.

10 Claims, 9 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 5

---------- KWIC ---------

Brief Summary Text - BSTX (5):

A **DMA** (Direct Memory Access) controller is a control device used for performing high-speed data **transfers** directly between peripheral units (e.g., a memory, an I/O (Input/Output) device) connected to a common bus without involving the CPU.

Brief Summary Text - BSTX (6):

FIGS. 5A to 5D are schematic diagrams each illustrating a conventional data **transfer** method using a **DMA** controller with each hexagonal block representing a single data **transfer** operation (e.g., a data **transfer** operation performed for a single word in one cycle). The data transfer operations are performed successively from left to right in the figures. The bus master at each data transfer operation is indicated inside the corresponding block. A bus master is a device which is controlling the current data transfer while occupying the bus. For example, the CPU or the DMA controller may be a bus master. In FIGS.

5A to 5D, the label "DMA" indicates that the DMA controller is occupying the bus as the bus master at the time, whereas "Other" indicates that a bus master other than the DMA controller (e.g., the CPU) is occupying the bus as the bus master at the time.

Brief Summary Text - BSTX (7):

FIG. 5A illustrates the "burst-mode **transfer** method" where the bus remains occupied by the **DMA** controller from activation of a **DMA transfer** operation to completion of the **DMA transfer** operation. Accordingly, during burst-mode **transfer** operations, another device (e.g., the CPU) must wait for a long time until the **DMA transfer** is completed before it may serve as the bus master and **transfer** data to and from the memory through the bus. In order to eliminate this long wait during burst-mode transfer operations, there have been proposed other data transfer methods as shown in FIGS. 5B, 5C and 5D.

Brief Summary Text - BSTX (8):

FIG. 5B illustrates a "word-by-word **transfer** method" wherein the bus master occupying the bus is forcibly switched between the **DMA** controller and another device after each one-word data **transfer**. FIG. 5C illustrates a "cycle-steal **transfer** method" where the **DMA** controller serves as the bus master for **DMA transfers** only when the bus is not occupied by the other device serving as the bus master. FIG. 5D illustrates a "timer interruption transfer method" where the bus master occupying the bus is switched between two devices by interrupts generated at predetermined intervals based on a timer. In the timer interruption **transfer** method, the **DMA** controller performing a **DMA transfer** is forced to discontinue the **DMA transfer** when the interrupt is generated after a predetermined period of time. The predetermined period of time runs from the activation of the **DMA transfer**. Once the **DMA transfer** is forced to discontinue, the other device serves as the bus master and occupies the bus. When another interrupt is generated after the predetermined time, the other device is forced to discontinue serving as the bus master and release the bus so that the **DMA** controller again can serve as the bus master and resume the interrupted **DMA transfer**.

Brief Summary Text - BSTX (9):

For data **transfers** (e.g., **DMA transfers**) in a system where a plurality of devices which can serve as bus masters share a common bus, there has been a demand for improving the **transfer** efficiency for both the **DMA** controller and the other devices.

Brief Summary Text - BSTX (10):

In the "word-by-word **transfer** method", however, the **DMA transfer** operations cannot be performed successively. Therefore, this method cannot be used to access a memory employing a high-speed transfer mode (e.g., the page mode of a DRAM). This results in a very poor **DMA transfer** efficiency.

Brief Summary Text - BSTX (11):

In the "cycle-steal **transfer** method", besides the above-noted problem, the other devices serving as bus masters may occupy the bus for a long time during which the **DMA** controller cannot access the bus. In such a case, the **DMA** controller has to wait for a long time until it is allowed to perform a **DMA transfer and, therefore, the DMA transfer** may not be completed within a predetermined period of time.

Brief Summary Text - BSTX (26):

Thus, the invention described herein makes possible the advantages of: (1) providing a method of transferring data where, a first device serving as the bus master can occupy the bus until a predetermined number of data items are successively and completely transferred (e.g, in a **DMA transfer**), while the other device does not have to wait for a long time to perform a data **transfer,** thereby improving data **transfer** efficiency for both devices; and (2) providing a bus master control device which also realizes the above effects.

Drawing Description Text - DRTX (4):

FIGS. 3A and 3B are schematic diagrams each illustrating an exemplary operation of a **DMA transfer** according to the example of the present invention.

Detailed Description Text - DETX (3):

FIG. 1 is a block diagram showing a configuration of an information processing unit 408. The information processing unit 408 includes a CPU 401, a peripheral unit 404, a DMA controller 11 and a bus controller 9. The CPU 401, the peripheral unit 404 and the bus controller 9 are connected to one another via an internal bus 406. The information processing unit 408 is connected to an external memory 405 via the bus controller 9 and an external bus 407. Reference numeral 301 denotes a CPU **transfer** request signal output to the bus controller 9 by the CPU 401 requesting occupation over the internal bus 406 and the external bus 407, whereas 101 denotes a **DMA transfer** activation request signal output to the **DMA** controller 11 by the peripheral unit 404 requesting a **DMA transfer**. As will be described later, the DMA controller 11 and the bus controller 9 are connected to each other via lines for carrying control signals and addresses.

Detailed Description Text - DETX (4):

The CPU 401 is a central processing unit for performing arithmetic operations and for controlling the whole process of the information processing unit 408. The peripheral unit 404 performs a specified group of processes based on instructions from the CPU 401. The peripheral unit 404 is typically an external memory device (e.g., a hard disk drive or an optical disk drive), a printer or the like. The bus controller 9 controls access to the internal bus 406 and the external bus 407, with the CPU 401 and the DMA controller 11 being devices which may serve as the bus master. When these devices request occupation over the internal bus 406 and the external bus 407, the bus controller 9 arbitrates between the devices 401 and 11 and determines which one of the devices 401 and 11 is to occupy the bus as the bus master. The **DMA** controller 11 controls data **transfers** (i.e., **DMA transfers**) performed directly between the peripheral unit 404 and the external memory 405 without involving the CPU 401 based on a request from the peripheral unit 404.

Detailed Description Text - DETX (6):

The DMA controller 11 includes a source address register 1a, a destination address register 1b and an address generation section 7. The source address register 1a stores a source address for a **DMA transfer** operation, whereas the destination address register 1b stores a destination address for a **DMA transfer** operation. An output signal from the source address register 1a and an output signal from the destination address register 1b are input to the address generation section 7. A read acknowledge signal 106 and a write acknowledge

signal 110 output from the bus controller 9 are input to the address generation section 7. The read acknowledge signal 106 indicates that the bus controller 9 has accepted a **DMA transfer** request and has begun to read out data to be transferred by a **DMA transfer**. The write acknowledge signals 110 indicates that the bus controller 9 has begun to write data to be transferred by a **DMA transfer**. The address generation section 7 reads out the contents of the source address register 1a and the destination address register 1b as data transfer start addresses, and outputs the data transfer start addresses to the bus controller 9 as a source address 105a and a destination address 105b. Then, based on the read acknowledge signal 106 and the write acknowledge signals 110, the address generation section 7 updates the address from which data is to be read out next and the address into which the data is to be written next, and outputs the updated addresses to the bus controller 9 as the source address 105a and the destination address 105b. Thus, a predetermined number of data items are transferred successively.

Detailed Description Text - DETX (7):

The **DMA** controller 11 further includes a **transfer** count register 2 and a subset **transfer** count register 3. The **transfer** count register 2 stores the number of **DMA transfer** operations to be performed. The subset transfer count register 3 stores a predetermined number as the number of data items to be transferred in a single subset transfer. The **DMA** controller 11 intermittently **transfers** all data items to be transferred by a **DMA transfer** by transferring a data subset including the predetermined number

Detailed Description Text - DETX (9):

The DMA controller 11 further includes counters 4 and 5 and a decrementor 6. The counter 4, which is reset at the beginning of a subset transfer, counts the read acknowledge signal 106 input from the bus controller 9. Thus, the counter 4 indicates the number of data items which have been read out from the respective source addresses out of the predetermined number of data items to be transferred in a single subset transfer. The counter 5, which is also reset at the beginning of the subset transfer, counts a transfer completion signal 102 input from the bus controller 9. The **transfer** completion signal 102 is output from the bus controller 9 indicating that a single **DMA transfer** operation for a single data item is completed. Thus, the counter 5 indicates the number of data items which have been written into the respective destination addresses (i.e., the number of data items for which the transfer operation has been completed) out of the predetermined number of data items to be transferred in a single subset transfer. The transfer completion signal 102 is also input to the transfer count register 2 and the decrementor 6. Each time the transfer completion signal 102 is asserted, the decrementor 6 decrements the output value from the transfer count register 2 by one and outputs the resultant value to the transfer count register 2. Thus, the number of data items which have been transferred out of the number of all the data items to be transferred is counted by the transfer count register 2 and the decrementor 6. Consequently, the number of the remaining data items to be transferred is stored in the transfer count register 2.

Detailed Description Text - DETX (10):

The **DMA** controller 11 further includes a **DMA transfer** control section 8 for controlling **DMA transfers**. The **DMA transfer** control section 8 receives as inputs the **DMA transfer** activation request signal 101, the read acknowledge signal 106, the write acknowledge signals 110, the **transfer** completion signal 102, an output signal 108 from the counter 4, an output signal 109 from the counter 5, an output signal 111 from the subset **transfer** count register 3 and an output signal 112 from the **transfer** count register 2, and the **DMA transfer**

control section 8 outputs an entire **transfer** completion signal 103 and a **DMA transfer** request signal 104. The **DMA transfer** control section 8 asserts the **DMA transfer** request signal 104 to the bus controller 9 in response to the **DMA transfer** activation request signal 101 from the peripheral unit 404, thus requesting bus occupation. Based on the output signal 108 from the counter 4, the output signal 109 from the counter 5 and the output signal 111 from the subset **transfer** count register 3, the **DMA transfer** control section 8 determines if all the predetermined number of data items to be transferred in a single subset **transfer** have been transferred. When it is determined that the predetermined number of data items have been transferred, the **DMA transfer** control section 8 terminates the assertion of the **DMA transfer** request signal 104 to the bus controller 9, thus instructing bus release. The **DMA transfer** control section 8 determines if all the data items to be transferred have been transferred based on the output signal 112 from the **transfer** count register 2. When it is determined that all the data items to be transferred have not been transferred, the **DMA transfer** control section 8 asserts the **DMA transfer** request signal 104 again to the bus controller 9 after completion of the current subset **transfer,** thus requesting to occupy the bus again and similarly control the subsequent subset **transfer**. When it is determined that all the data items to be transferred have been transferred, the **DMA transfer** control section 8 terminates the assertion of the **DMA transfer** request signal 104 to the bus controller 9, thus instructing bus release.

Detailed Description Text - DETX (11):

Based on the **DMA transfer** request signal 104 from the **DMA transfer** control section 8 and the CPU **transfer** request signal 301 from the CPU 401, the bus controller 9 arbitrates between the **DMA** controller 11 and the CPU 401 to determine which one of the devices 11 and 401 is to occupy the internal bus 406 and the external bus 407 as the bus master. When it is determined that the DMA controller 11 is to occupy these buses, the bus controller 9 outputs the source address 105a and the destination address 105b output from the address generation section 7 of the DMA controller 11 to the external bus 407 and internal bus 406, respectively, (or the source address 105a and the destination address 105b to the internal bus 406 and the external bus 407, respectively,) thus controlling read and write operations. The bus controller 9 outputs to the DMA controller 11 the read acknowledge signal 106 indicating that a read operation has begun, and the write acknowledge signal 110 indicating that a write operation has begun.

Detailed Description Text - DETX (14):

FIG. 3A shows a case where the CPU 401 occupies the bus for one cycle between each two subset **transfers** performed by the **DMA** controller 11. Herein, the period of bus occupation by the CPU 401 is not limited to one cycle as in FIG. 3A but, in fact, the CPU 401 may occupy the bus as long as required. Nevertheless, the CPU 401 typically needs to occupy the bus continuously for only a short time. Therefore, soon after the DMA controller 11 releases the bus to the CPU 401, the DMA controller 11 can regain bus occupation and start transferring a subset of four data items subsequent to the previous subset of four data items.

Detailed Description Text - DETX (15):

FIG. 3B shows a case where the CPU 401 occupies the bus for three cycles between the first and second subset transfers. Since the CPU 401 is not requesting bus occupation after the second subset **transfer, the DMA** controller 11 retains bus occupation and **transfers** the subsequent data subset.

Detailed Description Text - DETX (17):

FIG. 4 is a timing chart for illustrating the first subset transfer in FIGS. 3A and 3B. FIG. 4 shows the following signals for cycles t0 to t14 based on the clock, in this order: a clock signal which serves as the basis for operation timings for the information processing unit 408; the **DMA transfer** activation request signal 101; the **DMA transfer** request signal 104; the read acknowledge signal 106 output from the bus controller 9 to the **DMA transfer** control section 8 indicating acceptance of a **DMA transfer** request and beginning of a read operation; the output signal 108 from the counter 4; the write acknowledge signal 110 output from the bus controller 9 to the **DMA transfer** control section 8 indicating beginning of a write operation; the **transfer** completion signal 102 output from the bus controller 9 to the **DMA transfer** control section 8 indicating completion of a single **transfer** operation for transferring a single data item; the output signal 109 from the counter 5; the **transfer** count register 2; the source address 105a; the destination address 105b; the CPU **transfer** request signal 301; the external bus 407; and the internal bus 406.

Detailed Description Text - DETX (18):

It should be noted that the **DMA transfer** activation request signal 101 and the CPU **transfer** request signal 301 are active-high signals which are asserted when high, whereas the **DMA transfer** request signal 104, the read acknowledge signal 106, the write acknowledge signal 110 and the **transfer** completion signal 102 are active-low signals which are asserted when low.

Detailed Description Text - DETX (21):

First, based on an instruction from the CPU 401 or other external units, the **DMA transfer** control section 8 sets the source address register 1a to "1000" which is to be the source address, and the destination address register 1b to "2000" which is to be the destination address, and sets the **transfer** count register 2 and the subset **transfer** count register 3 to "12" and "4", respectively.

Detailed Description Text - DETX (23):

The peripheral unit 404 asserts the **DMA transfer** activation request signal 101 to the **DMA transfer** control section 8 of the **DMA** controller 11.

Detailed Description Text - DETX (25):

When detecting that the **DMA transfer** activation request signal 101 is high and asserted, the **DMA transfer** control section 8 outputs the **DMA transfer** request signal 104 (low) to the bus controller 9. Based on an instruction from the **DMA transfer** control section 8, the address generation section 7 reads out the values of the source address register 1a and the destination address register 1b, and outputs address #1000 as the source address 105a and address #2000 as the destination address 105b to the bus controller 9. The **DMA transfer** control section 8 also resets the counters 4 and 5.

Detailed Description Text - DETX (27):

When detecting that the **DMA transfer** request signal 104 from the **DMA transfer** control section 8 is low, the bus controller 9 determines to allow the **DMA** controller 11 to occupy the bus because the CPU **transfer** request signal 301 is not asserted. The bus controller 9 begins to read out data from address

#1000 in the external memory 405 through the external bus 407 and, simultaneously, outputs the read acknowledge signal 106 (low) to the **DMA transfer** control section 8.

Detailed Description Text - DETX (31):

Upon completion of the read operation from address #1000 through the external bus 407, the bus controller 9 begins to read out data from the next address #1001 in the external memory 405 and begins to write the data, which have been read out from address #1000 in the external memory 405, into address #2000 in the peripheral unit 404 through the internal bus 406. Simultaneously, the bus controller 9 outputs the read acknowledge signal 106 (low) and the write acknowledge signal 110 (low) to the **DMA transfer** control section 8.

Detailed Description Text - DETX (35):

Upon completion of the read operation from address #1001 through the external bus 407, the bus controller 9 begins to read out data from the next address #1002 in the external memory 405 and begins to sprite the data, which have been read out from address #1001 in the external memory 405, into address #2001 in the peripheral unit 404 through the internal bus 406. Simultaneously, the bus controller 9 outputs the read acknowledge signal 106 (low) and the write acknowledge signal 110 (low) to the **DMA transfer** control section 8. Upon completion of the write operation into address #2000 through the internal bus 406, the bus controller 9 sends the **transfer** completion signal 102 (low) to the **DMA transfer** control section 8.

Detailed Description Text - DETX (39):

Upon completion of the read operation from a address #1002 through the external bus 407, the bus controller 9 begins to read out data from the next address #1003 in the external memory 405 and begins to write the data, which have been read out from address #1002 in the external memory 405, into address #2002 in the peripheral unit 404 through the internal bus 406. Simultaneously, the bus controller 9 outputs the read acknowledge signal 106 (low) and the write acknowledge signal 110 (low) to the **DMA transfer** control section 8. Upon completion of the write operation into address #2001 through the internal bus 406, the bus controller 9 sends the **transfer** completion signal 102 (low) to the **DMA transfer** control section 8.

Detailed Description Text - DETX (43):

Upon completion of the read operation from address #1003 through the external bus 407, the bus controller 9 begins to write the data, which have been read out from address #1003 in the external memory 405, into address #2003 in the peripheral unit 404 through the internal bus 406. Simultaneously, the bus controller 9 outputs the write acknowledge signal 110 (low) to the **DMA transfer** control section 8. Upon completion of the write operation into address #2002 through the internal bus 406, the bus controller 9 sends the **transfer** completion signal 102 (low) to the **DMA transfer** control section 8.

Detailed Description Text - DETX (44):

When detecting that the signal 111 from the subset **transfer** count register 3 and the signal 108 from the counter 4, which have been input to **DMA transfer** control section 8, match each other both having the same value "4", the **DMA transfer** control section 8 determines that a data subset including four data items have been all read out from the respective source addresses, thus

terminating the assertion of the **DMA transfer** request signal 104 and instructing bus release.

Detailed Description Text - DETX (49):

Upon completion of the write operation into address #2003 through the internal bus 406, the bus controller 9 sends the **transfer** completion signal 102 (low) to the **DMA transfer** control section 8.

Detailed Description Text - DETX (52):

At this point of time, the CPU transfer request signal 301 from the CPU 401 is high, indicating that the CPU 401 is requesting bus occupation. As the bus controller 9 detects the CPU transfer request signal 301 from the CPU 401, the bus controller 9 performs an arbitration in bus occupation. Since the **DMA** controller 11 is not requesting bus occupation at this point of time with the **DMA transfer** request signal 104 being high, the bus controller 9 determines to allow the CPU 401 to occupy the bus at cycle t14.

Detailed Description Text - DETX (54):

When detecting that the signal 111 from the subset **transfer** count register 3 and the signal 109 from the counter 5, which have been input to the **DMA transfer** control section 8, match each other both having the same value "4", the **DMA transfer** control section 8 determines that a data subset including four data items have been all written into the respective destination addresses. The **DMA transfer** control section 8 again outputs the **DMA transfer** request signal 104 (low) to the bus controller 9 in order to perform the second subset **transfer**. However, the bus controller 9 has detected the CPU transfer request signal 301 from the CPU 401 at cycle t13 and, as a result of the bus occupation arbitration, the bus controller 9 has allowed the CPU 401 to occupy the bus as the bus master from cycle t14 and has already activated a data transfer operation. This data transfer operation corresponds to the CPU's transfer between the first and second subset transfers in FIG. 3A. Thus, the **DMA transfer** request signal 104 remains asserted by the **DMA** controller 11 and is accepted by the bus controller 9 after the CPU 401 releases the bus.

Detailed Description Text - DETX (55):

If there is no **transfer** request signal 301 from the CPU 401, the bus controller 9 accepts the **DMA transfer** request signal 104 at the cycle following cycle t14. In such a case, the **DMA** controller 11 consequently retains bus occupation without releasing the bus and performs the subsequent subset **transfer**.

Detailed Description Text - DETX (56):

The first subset transfer is performed as described above while reading out data through the external bus 407 from addresses #1000 to #1003 in the external memory 405 and writing the data through the internal bus 406 into respective addresses #2000 to #2003 in the peripheral unit 404. Subsequently, the second subset **transfer** shown in FIGS. 3A and 3B begins when the CPU 401 releases the bus and the bus controller 9 detects the **DMA transfer** request signal 104 which has been asserted since cycle t14. The second subset transfer is performed similarly as the first subset transfer while reading out data through the external bus 407 from addresses #1004 to #1007 in the external memory 405 and writing the data through the internal bus 406 into respective addresses #2004 to #2007 in the peripheral unit 404. Similarly, the third subset transfer is

performed while reading out data through the external bus 407 from addresses #1008 to #1011 in the external memory 405 and writing the data through the internal bus 406 into respective addresses #2008 to #2011 in the peripheral unit 404. At the time when the transfer completion signal 102 is asserted for the fourth time in the third subset transfer, the value of the transfer count register 2 is "0". When detecting that the output signal 112 from the **transfer** count register2 is "0", the **DMA transfer** control section 8 determines that all the data items to be transferred have been transferred and outputs to the bus controller 9 the entire **transfer** completion signal 103 for instructing bus release, thus completing the entire **transfer** operation.

Detailed Description Text - DETX (57):

When a **DMA** data **transfer** is interrupted by a DRAM refresh operation having a higher priority, the bus may become unavailable for data **transfer** operations for some cycles. However, since completion of a subset transfer is determined based on the number of data items which have been transferred but not a period of time, the number of data items to be transferred in a subset transfer is always ensured.

Detailed Description Text - DETX (60):

In the present example, the **DMA** controller 11 performs data **transfers on the intermittent** basis by transferring a data **subset** including a predetermined number of data items at a time. However, the **intermittent** data **transfers** may be performed by the CPU 401 while the **DMA** controller 11 **transfers** data during intervals between **subset transfer** operations performed by the CPU 401.

Detailed Description Text - DETX (61):

In the present example, completion of a single subset **transfer** is detected when it is determined that the data subset have been read out from the corresponding source addresses based on the signal 111 from the subset **transfer** count register 3 and the signal 108 from the counter 4, thus terminating the assertion of the **DMA transfer** request signal 104 and instructing bus release. However, it is also applicable to detect the completion of a single subset **transfer** when it is determined that the data subset read out from the corresponding source addresses have been written into the corresponding destination addresses based on the signal 111 from the subset **transfer** count register 3 and the signal 109 from the counter 5, thus terminating the assertion of the **DMA transfer** request signal 104 and instructing bus release.

Detailed Description Text - DETX (64):

Unlike the conventional techniques, the first device currently occupying the bus as the bus master is not forcibly switched to release the bus to a second device based on a predetermined period of time during which the first device is allowed to continuously occupy the bus. Therefore, the number of data items to be successively transferred in a single **transfer** operation can be ensured even when the period of time from the assertion of the **DMA transfer** activation request signal to the acceptance of the signal is not constant, or when a **DMA** data **transfer** is interrupted by a DRAM refresh operation having a higher priority so that the bus may become unavailable for data **transfers** for some cycles. In the present invention, the number of data items to be successively transferred in a single subset transfer is always ensured since completion of a data subset transfer is detected based on the number of data items which have been transferred but not on a time period. Data may be transferred more efficiently when each data transfer operation transfers a particular number of data items (e.g., 8 bytes) so as to better correspond to the data processing

procedures of the CPU or the peripheral unit. Moreover, data may be transferred even more efficiently when successively transferring, for example, 256-byte data being on the same page as in the high-speed page mode of DRAMs. This is particularly the case where the present invention is most effectively embodied.

PAT-NO: JP401230157A

DOCUMENT-IDENTIFIER: JP 01230157 A

TITLE: DMA CONTROLLER

PUBN-DATE: September 13, 1989

INVENTOR-INFORMATION:
NAME
ARAI, YOSHIHIDE

ASSIGNEE-INFORMATION:
NAME COUNTRY
RICOH CO LTD N/A

APPL-NO: JP63023598

APPL-DATE: February 3, 1988

INT-CL (IPC): G06F013/28, G06F012/00

ABSTRACT:

PURPOSE: To attain a **partial** segmenting **transfer** by generating the **partial** segmenting address of a **DMA** with a set value to a means which sets a **transfer** starting address, a **transfer** width and a blank width.

CONSTITUTION: When the transfer starting address, the transfer width and the blank width are respectively set at registers, a starting address (i) is outputted onto an S bus, a block end address (i) is outputted onto an E bus successively, and based on a result, a starting address (i)+1 is prepared on an N bus. Consequently, the **partial** segmenting address of the DMA can be generated at high speed with a simple control. Thus, with a simple constitution, the title **DMA** controller to generate the address, which can execute the **partial** segmenting **transfer,** can be obtained.

PUB-NO: EP000288649A1

DOCUMENT-IDENTIFIER: EP 288649 A1

TITLE: Memory control subsystem.

PUBN-DATE: November 2, 1988

INVENTOR-INFORMATION:

NAME	COUNTRY
GACH, ALAIN	N/A
HARTMANN, YVES	N/A
PEYPRONNENC, MICHEL	N/A

ASSIGNEE-INFORMATION:

NAME	COUNTRY
IBM	US

APPL-NO: EP87430014

APPL-DATE: April 22, 1987

PRIORITY-DATA: EP87430014A (April 22, 1987)

INT-CL (IPC): G06F013/34

EUR-CL (EPC): G06F013/16 ; G06F013/34, G06F012/08

US-CL-CURRENT: 710/22, 711/118

ABSTRACT:

CHG DATE=19990617 STATUS=O> The memory control subsystem controls and arbitrates the access to a memory 10 which is shared by a plurality of users comprising at least a processor 2 with its cache and input/output devices 4 having direct access to the memory through a direct memory access bus 12. It comprises a processor controller 20, a DMA controller 22 and a memory controller 24. A processor request is buffered into the processor controller 20 and is serviced right away if the memory controller is available, possibly with a simulataneous **transfer** between the devices 4 and buffers in the **DMA** controller 22. If the memory controller 24 is busy, because a **DMA** request is being serviced, the **DMA** controller comprises means to cause the **DMA transfer** to be interrupted, the processor request to be serviced and the **DMA transfer** to be resumed afterwards. Write requests made by the processor are buffered into processor controller 20 and an acknowledgement signal is sent to the processor which can resume execution without waiting the memory update completion. A read request which does not hit the cache is sent to the processor controller which causes the cache to be updated. In case of multiple processor requests contending with a long **DMA transfer,** the latter is sliced into several parts, each part mapping one cache line. In case of a DMA write, the cache lines which correspond to memory positions whose content is modified by the write operation are invalidated in such a way that the processor cannot read a **partially** written line into the cache.

DERWENT-ACC-NO: 1998-297345

DERWENT-WEEK: 199826

TITLE: DMA I/O devices supporting system on peripheral component interconnect bus - performs continuous DMA transfer as long as DMA I/O device continues to assert signal or as long as DMA controller is programmed

INVENTOR: JIRGAL, J J

PATENT-ASSIGNEE: VLSI TECHNOLOGY INC[VLSIN]

PRIORITY-DATA: 1995US-0488989 (June 8, 1995)

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
US 5752081 A	May 12, 1998	N/A	008	G06F 015/16

APPLICATION-DATA:

PUB-NO	APPL-DESCRIPTOR	APPL-NO	APPL-DATE
US 5752081A	N/A	1995US-0488989	June 8, 1995

INT-CL (IPC): G06F013/00, G06F013/28 , G06F015/16

ABSTRACTED-PUB-NO: US 5752081A

BASIC-ABSTRACT:

The system (10) includes a serial link signalling unit connected to at least one DMA I/O device (12). The DMA I/O device is directly connected to a PCI bus (14) for requesting DMA transfer from or to a memory. A DMA controller (16) is connected to the PCI bus and to the DMA I/O device for signal reception. The controller permits transfer of data to the DMA I/O device from the memory or from the DMA I/O device to the memory. The signalling unit notifies the DMA controller for indicating requirement for DMA transfer. An arbiter (24) is coupled to the DMA controller for receiving signal indicating DMA transfer request. A first signal line (32) is connected to the DMA I/O device and DMA controller for signalling the DMA I/O device, when DMA transfer is granted.

A code indicating active DMA channels acting as **subset** of PCI address lines, is also sent to the DMA I/O device through the first signal line. A second signal line (34) is connected to the **DMA** I/O device and the controller, for signalling the controller when acknowledgement is received from the corresponding **DMA transfer** granted **DMA** I/O device. The **transfer** is continued as long as the **DMA** I/O device continues to assert the signal or as long as the **DMA** controller is programmed.

ADVANTAGE - Prevents data overrun and underrun, by indicating generation of request signal at real time. Reduces cost by reducing required number of signal pins of bus.

CHOSEN-DRAWING: Dwg.1/3

TITLE-TERMS: DMA DEVICE SUPPORT SYSTEM PERIPHERAL COMPONENT INTERCONNECT BUS PERFORMANCE CONTINUOUS DMA TRANSFER LONG DMA DEVICE CONTINUE SIGNAL

LONG DMA CONTROL PROGRAM

DERWENT-CLASS: T01

EPI-CODES: T01-H01A; T01-H01B1; T01-H05B2;

SECONDARY-ACC-NO:
Non-CPI Secondary Accession Numbers: N1998-232689

US-PAT-NO: 5481678

DOCUMENT-IDENTIFIER: US 5481678 A

TITLE: Data processor including selection mechanism for coupling internal and external request signals to interrupt and DMA controllers

DATE-ISSUED: January 2, 1996

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Kondo; Hiroyuki	Itami	N/A	N/A	JP
Nakao; Yuichi	Itami	N/A	N/A	JP
Koyama; Kazumi	Itami	N/A	N/A	JP

US-CL-CURRENT: 710/22, 710/28

ABSTRACT:

A data processor includes an interrupt control unit (102); a direct memory access control unit (103); a selection unit (104) for selecting a connection of ready and error signals from a serial communication control unit and external interrupt and DMA request signals with interrupt inputs to the interrupt and the DMA control units; and a selection setting unit (105) for setting a selection mode of the selection unit, thereby making possible to not only switch between a process by interrupt and a process by DMA transfer by programming according to the application but also make efficient use of the hardware resources in the data processor.

10 Claims, 8 Drawing figures

Exemplary Claim Number: 8

Number of Drawing Sheets: 7

---------- KWIC ---------

Claims Text - CLTX (39):

configuring said selection circuit, responsive to said serial communications mode, to route particular ones of the communication control signals from the serial communications controller to selected ones of the one or more interrupt request lines of the interrupt controller and the one or more **DMA transfer** request lines of the **DMA** controller, said selection circuit also routing, depending upon said communications mode, a first **subset** of said first plurality of interrupt request signals to those ones of the one or more interrupt request lines of the interrupt controller not coupled to any one of the communication control signals from the serial communications controller and a second **subset** of said second plurality of **DMA** request signals to those ones of the one or more **DMA transfer** request lines of the **DMA** controller not coupled to any one of the communication control signals from the serial communications controller.

L Number	Hits	Search Text	DB	Time stamp
1	6573	dma with transfer$1	USPAT; US-PGPUB	2003/05/13 19:09
4	80	subset$1 same intermittent	USPAT; US-PGPUB	2003/05/13 19:09
7	4	(subset$1 same intermittent) and (dma with transfer$1)	USPAT; US-PGPUB	2003/05/13 19:07
10	3772	dma with transfer$1	EPO; JPO; DERWENT; IBM_TDB	2003/05/13 19:09
15	6	subset$1 same intermittent	EPO; JPO; DERWENT; IBM_TDB	2003/05/13 19:09
20	0	(dma with transfer$1) and (subset$1 same intermittent)	EPO; JPO; DERWENT; IBM_TDB	2003/05/13 19:09
25	29	(subset$1 or partial$3) same (dma with transfer$1)	EPO; JPO; DERWENT; IBM_TDB	2003/05/13 19:15
30	144	(subset$1 or partial$3) same (dma with transfer$1)	USPAT; US-PGPUB	2003/05/13 19:16
33	11	30.ti,clm,ab.	USPAT; US-PGPUB	2003/05/13 19:16

PGPUB-DOCUMENT-NUMBER: 20030046514

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20030046514 A1

TITLE: Single-chip microcomputer

PUBLICATION-DATE: March 6, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY
RULE-47			
Kawasaki, Shumpei	Tokyo		JP
Akao, Yasushi	Tokyo		JP
Noguchi, Kouki	Tokyo		JP
Hasegawa, Atsushi	Tokyo		JP
Ohsuga, Hiroshi	Tokyo		JP
Kurakazu, Keiichi	Tokorozawa-shi		JP
Matsubara, Kiyoshi	Toyko		JP
Hayakawa, Akio	Tokyo		JP
Ito, Yoshitaka	Tokyo		JP

US-CL-CURRENT: 712/33

ABSTRACT:

A single-chip microcomputer comprising: a first bus having a central processing unit and a cache memory connected therewith; a second bus having a dynamic memory access control circuit and an external bus interface connected therewith; a break controller for connecting the first bus and the second bus selectively; a third bus having a peripheral module connected therewith and having a lower-speed bus cycle than the bus cycles of the first and second buses; and a bus state controller for effecting a data transfer and a synchronization between the second bus and the third bus. The single-chip microcomputer has the three divided internal buses to reduce the load capacity upon the signal transmission paths so that the signal transmission can be accomplished at a high speed. Moreover, the peripheral module required to have no operation speed is isolated so that the power dissipation can be reduced.

---------- KWIC ---------

Detail Description Paragraph - DETX (27):

[0111] For the DMAC, external terminals /DREQ0 and /DREQ1 are input terminals to be fed with a direct memory access (DMA) **transfer request** signal from an external device and correspond to the channel 1 and the channel 2. External terminals /DACK0 and /DACK1 are output terminals for outputting a DMA transfer accept (i.e., DMA transfer acknowledge) signal to the external device having outputted the DMA **transfer request** signal, when the DMA transfer is acknowledged to the DMA **transfer request** signal, and correspond to the channel 1 and the channel 2.

Detail Description Paragraph - DETX (107):

[0191] The bus mode (or transfer mode) is divided into a **cycle steal** mode and a burst mode. In the **cycle steal** mode, after the end of **DMA transfer** of one word, the bus right is released and transferred to another bus master

(e.g., CPU). In the burst mode, when a bus right is acquired, the **DMA transfer** is continued till the **transfer** ending condition is satisfied. In case, however, the level of the terminal /DREQ is sampled in the external request mode, the **DMA transfer** is executed according to the level of the terminal /DREQ.

Detail Description Paragraph - DETX (108):

[0192] A **transfer request** is divided into an external request, an internal request from an internal peripheral module, and an auto-request. The external request can start the channel CH0 by the terminal /DREQ0 and the channel CH1 by the terminal /DREQ1. In the sampling of the terminals /DREQ0 and /DREQ1, the fall edge and level can be selected as the select conditions. The internal request from the internal peripheral module is divided into the receive data full of the internal SCI0 and the send data line empty of the internal SCI. These requests are automatically cleared by starting the DMA transfer cycle. For the auto-request, the transfer operation is started by setting the DE bits of the channel control registers DHCRn (0, 1) of the DMAC.

Detail Description Paragraph - DETX (109):

[0193] In case a plurality of channel **transfer requests** are simultaneously made for the DMAC, the transfer channel is determined in accordance with the priority order. This priority order is divided into a priority order fixed mode and an alternate mode. In the priority order fixed mode, the priority order between the individual channels is unchanged. Two channels of 1 or 0 can be fixed to the priority order. In the alternate mode, the priority order is alternated between the channels CH0 and CH1, and the timing for changing the priority order is determined when one transfer unit (byte or word) of the channel CH0 or CH1 is ended by the round robin scheduling.

US-PAT-NO: 6429871

DOCUMENT-IDENTIFIER: US 6429871 B1

TITLE: Graphic processing method and system for displaying a combination of images

DATE-ISSUED: August 6, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Katsura; Koyo	Hitachiota	N/A	N/A	JP
Matsuo; Shigeru	Hitachi	N/A	N/A	JP
Sato; Jun	Musashino	N/A	N/A	JP
Sone; Takashi	Tokyo	N/A	N/A	JP
Yokota; Yoshikazu	Kodaira	N/A	N/A	JP
Kikuchi; Masahiko	Hitachi	N/A	N/A	JP

US-CL-CURRENT: 345/501, 345/213 , 345/505 , 345/629

ABSTRACT:

A graphic processing method and system which is capable of displaying a combination of images from, for example, an external source such as TV signals and graphic data generated by a graphic processor, whereby the graphic processor fetches horizontal synchronizing signals and vertical synchronizing signals, via respective terminals, from a source external of the system, and reads out the data from a frame buffer in accordance with the fetched signals to display the data.

2 Claims, 84 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 69

---------- KWIC ---------

Detailed Description Text - DETX (40):

(8) DMA **transfer request** (DREQ: Output)

Detailed Description Text - DETX (41):

Output signal to request a data **transfer** to the DMAC 13 when a data **transfer** is achieved in the **DMA transfer** mode. For the **DMA transfer** operation, a **cycle steal** or a burst mode can be selected.

Detailed Description Text - DETX (137):

Incidentally, since the **transfer** is executed in the **cycle steal** mode regardless of the setting of the DRC, any register in the graphic display processor (GDP) 10 can be accessed from the side of the central processing unit (CPU) 11; consequently, the command **DMA transfer** can be stopped by clearing the CDM to "0" or by use of a DONE input.

Detailed Description Text - DETX (138):

DMA transfer request control (DRC) DRC 0 Sends the DREQ in the form of a level signal (burst mode). However, this bit can be set only by an execution of a data **DMA transfer** command. 1 Sends the DREQ in the form of pulse signals for each transfer of a word (byte) (**cycle steal** mode).

US-PAT-NO: 6279063

DOCUMENT-IDENTIFIER: US 6279063 B1

TITLE: Microcomputer system with at least first and second microcomputers each operable in master and slave modes with configurable bus access control terminals and bus use priority controller

DATE-ISSUED: August 21, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Kawasaki; Shumpei	Tokyo	N/A	N/A	JP
Akao; Yasushi	Kokubunji	N/A	N/A	JP
Noguchi; Kouki	Tokyo	N/A	N/A	JP
Hasegawa; Atsushi	Tachikawa	N/A	N/A	JP
Ohsuga; Hiroshi	Hino	N/A	N/A	JP
Kurakazu; Keiichi	Tokorozawa	N/A	N/A	JP
Matsubara; Kiyoshi	Higashimurayama	N/A	N/A	JP
Hayakawa; Akio	Hachiouji	N/A	N/A	JP
Ito; Yoshitaka	Kodaira	N/A	N/A	JP

US-CL-CURRENT: 710/110, 712/229 , 712/31 , 712/32 , 712/43

ABSTRACT:

A single-chip microcomputer comprising: a first bus having a central processing unit and a cache memory connected therewith; a second bus having a dynamic memory access control circuit and an external bus interface connected therewith; a break controller for connecting the first bus and the second bus selectively; a third bus having a peripheral module connected therewith and having a lower-speed bus cycle than the bus cycles of the first and second buses; and a bus state controller for effecting a data transfer and a synchronization between the second bus and the third bus. The single-chip microcomputer has the three divided internal buses to reduce the load capacity upon the signal transmission paths so that the signal transmission can be accomplished at a high speed. Moreover, the peripheral module required to have no operation speed is isolated so that the power dissipation can be reduced.

9 Claims, 44 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 42

---------- KWIC ---------

Detailed Description Text - DETX (27):

For the DMAC, external terminals /DREQ0 and /DREQ1 are input terminals to be fed with a direct memory access (DMA) **transfer request** signal from an external device and correspond to the channel 1 and the channel 2. External terminals /DACK0 and /DACK1 are output terminals for outputting a DMA transfer accept (i.e., DMA transfer acknowledge) signal to the external device having outputted the DMA **transfer request** signal, when the DMA transfer is acknowledged to the DMA **transfer request** signal, and correspond to the channel 1 and the channel 2.

Detailed Description Text - DETX (107):

The bus mode (or transfer mode) is divided into a **cycle steal** mode and a burst mode. In the **cycle steal** mode, after the end of **DMA transfer** of one word, the bus right is released and transferred to another bus master (e.g., CPU). In the burst mode, when a bus right is acquired, the **DMA transfer** is continued till the **transfer** ending condition is satisfied. In case, however, the level of the terminal /DREQ is sampled in the external request mode, the **DMA transfer** is executed according to the level of the terminal /DREQ.

Detailed Description Text - DETX (108):

A **transfer request** is divided into an external request, an internal request from an internal peripheral module, and an auto-request. The external request can start the channel CH0 by the terminal /DREQO and the channel CH1 by the terminal /DREQ1. In the sampling of the terminals /DREQ0 and /DREQ1, the fall edge and level can be selected as the select conditions. The internal request from the internal peripheral module is divided into the receive data full of the internal SCI0 and the send data line empty of the internal SCI. These requests are automatically cleared by starting the DMA transfer cycle. For the auto-request, the transfer operation is started by setting the DE bits of the channel control registers DHCRn (0, 1) of the DMAC.

Detailed Description Text - DETX (109):

In case a plurality of channel **transfer requests** are simultaneously made for the DMAC, the transfer channel is determined in accordance with the priority order. This priority order is divided into a priority order fixed mode and an alternate mode. In the priority order fixed mode, the priority order between the individual channels is unchanged. Two channels of 1 or 0 can be fixed to the priority order. In the alternate mode, the priority order is alternated between the channels CH0 and CH1, and the timing for changing the priority order is determined when one transfer unit (byte or word) of the channel CH0 or CH1 is ended by the round robin scheduling.

US-PAT-NO: 6249833

DOCUMENT-IDENTIFIER: US 6249833 B1

TITLE: Dual bus processing apparatus wherein second control means request access of first data bus from first control means while occupying second data bus

DATE-ISSUED: June 19, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Takahashi; Junichi	Tokyo	N/A	N/A	JP

US-CL-CURRENT: 710/308, 710/100 , 710/107 , 710/27

ABSTRACT:

In an information processing apparatus equipped with a CPU, an operating rate of this CPU is increased so as to increase a throughput of this entire information processing apparatus. The information processing apparatus is arranged by first and second internal buses independently provided from each other, an internal memory connected to the first internal bus, and a timer 25 connected to the second internal bus. Furthermore, this information processing apparatus is arranged by an A/D converter, first/second serial interfaces, the CPU, and a DMAC (direct memory access controller). Both the CPU and the DMAC control data input/output operations in the internal memory and the timer while occupying at least one of these fist/second data buses. The DMAC supplies a request signal to such a CPU for controlling the data input/output operation of the internal memory while occupying at least one of the first/second internal buses, and also controls the data input/output operations in the internal memory in response to an acknowledge signal supplied from the CPU while occupying either one or both the first/second internal buses.

46 Claims, 27 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 22

---------- KWIC ---------

Detailed Description Text - DETX (16):

The channel validity signal TCA may be invalidated by a channel clear signal CHC supplied from the DMA cycle sequencer 38. It should be noted that a term "priority order" implies a channel priority order among a zero-th channel 43 to a third channel 46. The zero-th channel 43 owns the highest priority order, the first channel 44 and the second channel 45 own the priority orders lower than the highest priority order, and the third channel 46 owns the lowest priority order. As a consequence, there are no priority orders among the internal request signal IREQ, the external request signal EREQ, and the software **transfer request** signal SREQ. That is, a first signal owns a top priority order. Alternatively, priority orders may be set among these signals with respect to each of these channels.

Detailed Description Text - DETX (26):

As indicated in FIG. 6, the above-described DMA control register 35 is mainly arranged by 4 sets of channels, i.e, a zero-th channel 43 to a 3rd channel 46; two pieces of selectors 47 and 48; and an adding/subtracting device 49. Various sorts of data such as the source address SAD may be read/written via the internal bus 30 from/into the zero-th channel 43 through the 3rd channel 46 under control of the CPU 22. DMA address signals DMAD.sub.0 to DMAD.sub.3 outputted from the zero-th channel 43 to the 3rd channel 46 are entered into the selector 47. Among these DMA address signals, such an address signal selected by the channel validity signal TCA is outputted as a DMA address signal DMAD from this selector 47, and then this DMA address signal DMAD is supplied to the channel control unit 32 and the adding/subtracting device 49. The adding/subtracting device 49 adds/subtracts the DMA address signal DMAD supplied from the selector 47 and the displacement signal DPL supplied from the channel control unit 32 to thereby output a calculation result COUT. The calculation result COUT is written into the corresponding register of the corresponding channel at a falling edge of an address counter read strobe ACR. Also, DMA control signals DMC.sub.0 to DMC.sub.3 outputted from the zero-th channel 43 to the 3rd channel 46 are entered into the selector 48. Among these DMA control signals, such a control signal selected by the channel validity signal TCA is outputted as a DMA control signal DMC from this selector 48, and then this DMA control signal DMC is supplied to the channel control unit 32. Furthermore, software **transfer request** signals SREQ.sub.0 to SREQ.sub.3 outputted from the zero-th channel 43 to the 3rd channel 46 directly constitute a 4-bit software **transfer request** signal SREQ and then this 4-bit software **transfer request** signal SREQ is supplied to the channel control unit 32.

Detailed Description Text - DETX (31):

In the block transfer mode, once the DMAC 23 accepts the DMA **transfer request,** the DMAC 23 executes the DMA transfer operation of this channel until the byte count value of the valid channel becomes 0, during which even when a DMA transfer demand for a channel having a higher priority than that of the present channel is issued, the present channel is not changed, but also the byte cycle of the CPU is not interrupted. When the byte count value becomes 0, the DMA cycle sequencer 38 outputs the channel clear signal CHC.

Detailed Description Text - DETX (32):

The software transfer demand constitutes as a software **transfer request** signal SREQ.sub.0, a 4-bit software **transfer request** signal SREQ in combination with the software **transfer request** signals SREQ.sub.1 to SREQ.sub.3 supplied from other channels, namely first channel 44 to third channel 46. Then, this software **transfer request** signal SREQ.sub.0 is supplied to the channel control unit 32.

Detailed Description Text - DETX (53):

In the above-described operation, the process operation defined at the step SP4 is related to one of the **DMA transfer** systems called as a "CPU **cycle steal**". However, the present invention is apparently not limited to this CPU **cycle steal** system. Alternatively, for example, the information processing apparatus of the present invention may employ other **DMA transfer** systems such as the memory **cycle steal** system where the data is transferred by utilizing the empty time of the internal memory 24, and the interlock **transfer** system where the operation of the CPU 22 is stopped by entering the hold input, and the data is transferred while stopping the operation of the CPU 22.

Detailed Description Text - DETX (69):

In the above-explained operations, the internal request signal is inputted from a single I/O device into the DMAC 23. Next, various operations will be described in such a case that a plurality of internal request signals IREQs, the external request signal EREQ, or the software **transfer request** signal SREQ are inputted to the DMAC 23.

Detailed Description Text - DETX (70):

The above-described internal request signal IREQ, external request signal EREQ, and software **transfer request** signal SREQ are OR-gated by the OR gate 36 of the channel control unit 32 for constituting the DMAC 23 to produce a 4-bit bus request signal BREQ which will then be supplied to the priority encoder 37 (see FIG. 3). It should be noted that the respective bits BREQ.sub.0 to BREQ.sub.3 of the bus request signal BREQ correspond to the zero-th channel 43 through the 3rd channel 46. The zero-th channel 43 owns the highest priority, the first channel 44 owns the second highest priority, and the second channel 45 owns the third highest priority, and further, the third channel 46 owns the lowest priority. As a consequence, as represented in FIG. 13(1) to FIG. 13(4), when the respective bits BREQ.sub.0 to BREQ.sub.3 of the bus request signal BREQ are changed, the priority encoder 37 changes the respective bits TCA.sub.0 to TCA.sub.3 of the channel validity signal TCA in accordance with the above-described priority orders as represented in FIG. 13(5) to FIG. 13(8), and then outputs the channel validity signal TAC having the changed bits.

Detailed Description Text - DETX (111):

When the CPU 62 fetches a command code of the printer control program stored in the first external memory 66 after releasing the system reset, the CPU 62 commences the printer control. First, the CPU 62 DMA-transfers the outline font stored in the second external memory 67 to the internal memory 24. In this case, the CPU 62 transfers the address "p" of the second external memory 67 equal to a source as a source address SAD via the internal bus 30 to the DMAC 63. The CPU 62 transfers the address "u" of the internal memory 24 equal to a destination as a destination address DAD via the internal bus 30 to the DMAC 63. The CPU 62 transfers a transfer number corresponding to a size of outline font data which should be transferred as the byte count value BC via the internal bus 30 to the DMAC 63. The CPU 62 transfers other software **transfer requests,** the type/mode of DMA transfer operation, the displacement value ".alpha.", and the value ".beta." to be subtracted from the byte count value BC via the internal bus 30 to the DMAC 63.

Detailed Description Text - DETX (118):

Next, the bitmap data expanded in the internal memory 24 is DMA-transferred to the second serial interface 28. Also, in this case, the CPU 62 transfers via the internal bus 30 to the DMAC 63, the address of the internal memory 24 corresponding to the source as the source address, where the bitmap data is stored. The CPU 62 transfers the address "s" of the second serial control register for constituting the second serial interface 28 equal to the destination as the destination address DAD via the internal bus 30 to the DMAC 63. The CPU 62 transfers a transfer number corresponding to an amount of bitmap data which should be transferred as the byte count value BC via the internal bus 30 to the DMAC 63. The CPU 62 transfers other software **transfer requests,** the type/mode of DMA transfer operation, the displacement value ".alpha.", and the value ".beta." to be subtracted from the byte count value BC via the internal bus 30 to the DMAC 63.

Detailed Description Text - DETX (122):

In the above-explained second embodiment, one software **transfer request** signal SREQ is inputted into the DMAC 63. When a plurality of internal request signals IREQs, a plurality of external request signals EREQs, or a plurality of software **transfer request** signals EREQs are entered into the DMAC 63, operations in accordance with the priority orders thereof are similar to those of the first embodiment. Therefore, a detailed description of the operations according to the second embodiment is omitted. Although the second embodiment does not specifically describe operations in the various types of DMA transfer operations and the various modes thereof, since these operations are similar to those of the first embodiment, a detailed description thereof is omitted.

Detailed Description Text - DETX (141):

Furthermore, in the above-described embodiments, the internal request signal IREQ, the external request signal EREQ, and the software **transfer request** signal SREQ are supplied as the bus request signal BREQ via the OR gate 36 to the priority encoder 37 shown in FIG. 3 and FIG. 20. As a result, the priority encoder 73 cannot judge which sort of request signal is supplied. The present invention is not limited to this circuit arrangement. Alternatively, for instance, priority orders may be separately set to the internal request signal IREQ, the external request signal EREQ, and the software **transfer request** signal SREQ. In such an alternative case that plural different sorts of request signals are supplied within a preselected time period, such a selection means capable of selectively outputting a request signal with a higher priority may be provided instead of the OR gate 36.

US-PAT-NO: 6223265

DOCUMENT-IDENTIFIER: US 6223265 B1

TITLE: Single-chip microcomputer synchronously controlling external synchronous memory responsive to memory clock signal and clock enable signal

DATE-ISSUED: April 24, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Kawasaki; Shumpei	Tokyo	N/A	N/A	JP
Akao; Yasushi	Kokubunji	N/A	N/A	JP
Noguchi; Kouki	Tokyo	N/A	N/A	JP
Hasegawa; Atsushi	Tachikawa	N/A	N/A	JP
Ohsuga; Hiroshi	Hino	N/A	N/A	JP
Kurakazu; Keiichi	Tokorozawa	N/A	N/A	JP
Matsubara; Kiyoshi	Higashimurayama	N/A	N/A	JP
Hayakawa; Akio	Hachiouji	N/A	N/A	JP
Ito; Yoshitaka	Kodaira	N/A	N/A	JP

US-CL-CURRENT: 711/167, 711/105 , 711/111 , 712/33 , 712/34 , 712/35 , 712/36 , 712/37 , 712/38 , 712/39 , 712/40 , 713/500 , 713/600

ABSTRACT:

A single-chip microcomputer comprising: a first bus having a central processing unit and a cache memory connected therewith; a second bus having a dynamic memory access control circuit and an external bus interface connected therewith; a break controller for connecting the first bus and the second bus selectively; a third bus having a peripheral module connected therewith and having a lower-speed bus cycle than the bus cycles of the first and second buses; and a bus state controller for effecting a data transfer and a synchronization between the second bus and the third bus. The single-chip microcomputer has the three divided internal buses to reduce the load capacity upon the signal transmission paths so that the signal transmission can be accomplished at a high speed. Moreover, the peripheral module required to have no operation speed is isolated so that the power dissipation can be reduced.

19 Claims, 44 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 42

---------- KWIC ---------

Detailed Description Text - DETX (27):

For the DMAC, external terminals /DREQ0 and /DREQ1 are input terminals to be fed with a direct memory access (DMA) **transfer request** signal from an external device and correspond to the channel 1 and the channel 2. External terminals /DACK0 and /DACK1 are output terminals for outputting a DMA transfer accept (i.e., DMA transfer acknowledge) signal to the external device having outputted the DMA **transfer request** signal, when the DMA transfer is acknowledged to the

DMA **transfer request** signal, and correspond to the channel 1 and the channel 2.

Detailed Description Text - DETX (107):

The bus mode (or transfer mode) is divided into a **cycle steal** mode and a burst mode. In the **cycle steal** mode, after the end of **DMA transfer** of one word, the bus right is released and transferred to another bus master (e.g., CPU). In the burst mode, when a bus right is acquired, the **DMA transfer** is continued till the **transfer** ending condition is satisfied. In case, however, the level of the terminal /DREQ is sampled in the external request mode, the **DMA transfer** is executed according to the level of the terminal /DREQ.

Detailed Description Text - DETX (108):

A **transfer request** is divided into an external request, an internal request from an internal peripheral module, and an auto-request. The external request can start the channel CH0 by the terminal /DREQ0 and the channel CH1 by the terminal /DREQ1. In the sampling of the terminals /DREQ0 and /DREQ1, the fall edge and level can be selected as the select conditions. The internal request from the internal peripheral module is divided into the receive data full of the internal SCI0 and the send data line empty of the internal SCI. These requests are automatically cleared by starting the DMA transfer cycle. For the auto-request, the transfer operation is started by setting the DE bits of the channel control registers DHCRn (0, 1) of the DMAC.

Detailed Description Text - DETX (109):

In case a plurality of channel **transfer requests** are simultaneously made for the DMAC, the transfer channel is determined in accordance with the priority order. This priority order is divided into a priority order fixed mode and an alternate mode. In the priority order fixed mode, the priority order between the individual channels is unchanged. Two channels of 1 or 0 can be fixed to the priority order. In the alternate mode, the priority order is alternated between the channels CH0 and CH1, and the timing for changing the priority order is determined when one transfer unit (byte or word) of the channel CH0 or CH1 is ended by the round robin scheduling.

US-PAT-NO: 6131133

DOCUMENT-IDENTIFIER: US 6131133 A

TITLE: Data exchange interface that directly transmits control signals either to a microprocessor or a D.M.A. controller via a first and second control line respectively

DATE-ISSUED: October 10, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Salbaum; Helmut	Nurnberg	N/A	N/A	DE
Mehling; Rainer	Nurnberg	N/A	N/A	DE

US-CL-CURRENT: 710/48, 710/107 , 710/23

ABSTRACT:

A communication system includes a data exchange interface which is coupled via an internal bus to a microprocessor and a DMA unit. For controlling the data exchange, the interface transmits control signals either to a microprocessor or a DMA unit. The control signals are acknowledged each time. For realizing an efficient data exchange via the interface, the interface is an I.sup.2 C bus interface with minimum load of the microprocessor, and transmits a first group of control signals by a first control line to the DMA unit and a second group of control signals by a second control line to the microprocessor. By respective settings of the interface, a selectable part of the control circuitry may be transferred from the microprocessor to the DMA unit. The total number and type of control signals (interrupts) which correspond each to a specific mode of the interface may be retained.

8 Claims, 1 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 1

---------- KWIC ---------

Detailed Description Text - DETX (3):

Data are transmitted by the I.sup.2 C bus interface 4 and the I.sup.2 C bus 5 from and to electronic modules (ICs) which have an I.sup.2 C bus interface while support is given by the CPU 2 and the DMA unit 3. For this purpose, the interface 4 announces with each control signal (interrupt) by the control line 8 when data are received which are sent to the interface 4 by another module. The ICU 11 produces an interrupt vector for each control signal, which interrupt vector is transmitted to the core of the CPU 2. The CPU 2 evaluates the interrupt vector and starts a respective interrupt routine via the start address contained in the interrupt vector. This initializes the data exchange and a memory area in the RAM 7 is addressed in which the received data are stored. With large data sequences such as used, for example, for a graphics display, after the initialization the interface 4 transmits a control signal by the control line 10 to the DMA unit 3 which control signal corresponds to a request for a DMA transfer. The DMA transfer makes it possible to realize a rapid transmission of a received byte by the data bus 61 to the RAM 7, without

loading the CPU 2. After the transmission of the byte, the DMA unit 3 transmits a control signal to the interface 4 by the control line 9, which transmission acknowledges that the DMA **transfer request** is deleted. This program run is repeated with the next byte received by the interface 4 from the I.sup.2 C bus. When bytes are transmitted on the I.sup.2 C bus by the interface 4, the bytes buffered in the RAM 7 are transmitted accordingly by the data bus 61.

Detailed Description Text - DETX (4):

In this manner, the load of the CPU 2 is greatly reduced because it is no longer necessary for the CPU 2 to react to every control signal (interrupt) from the I.sup.2 C bus interface 4, but the control signals occurring most are sent from the interface 4 to the DMA unit 3 and processed there. Since an interrupt source (I.sup.2 C bus interface 4) now uses two control lines, an interrupt is processed and responded either by the CPU 2 or by the DMA unit 3 depending on the mode. With the conventional interrupt control of the I.sup.2 C bus interface 4 by the CPU 2, about 150 clock cycles are necessary for each byte to be transferred (the number of clock cycles relates to a 68,000 CPU). When the I.sup.2 C bus interface according to the invention is used, only about 10 clock cycles are necessary for each byte to be transferred. The **transfer by the DMA** unit 3 provides that there is no interrupt load of the CPU 2 during the **transfer,** but only the small number of interrupts for the start and stop of the **transfer**. When, in addition, the DMA unit 3 needs to have the 10 necessary clock cycles in the **cycle-steal** mode (this is, the CPU 2 does not need the internal buses 61 and 62 and makes the access by other connected function components possible), this leads to an optimum load of data bus 61 and address bus 62. Such interleaving of the bus access by DMA unit 3 and CPU 2 may provide that a data rate of 400 kbit/s can be processed on the I.sup.2 C bus 5 of the communication system 1.

US-PAT-NO: 5930523

DOCUMENT-IDENTIFIER: US 5930523 A

TITLE: Microcomputer having multiple bus structure coupling CPU to other processing elements

DATE-ISSUED: July 27, 1999

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Kawasaki; Shumpei	Tokyo	N/A	N/A	JP
Akao; Yasushi	Kokubunji	N/A	N/A	JP
Noguchi; Kouki	Tokyo	N/A	N/A	JP
Hasegawa; Atsushi	Tachikawa	N/A	N/A	JP
Ohsuga; Hiroshi	Hino	N/A	N/A	JP
Kurakazu; Keiichi	Tokorozawa	N/A	N/A	JP
Matsubara; Kiyoshi	Higashimurayama	N/A	N/A	JP
Hayakawa; Akio	Hachiouji	N/A	N/A	JP
Ito; Yoshitaka	Kodaira	N/A	N/A	JP

US-CL-CURRENT: 712/32, 710/100

ABSTRACT:

A single-chip microcomputer comprising: a first bus having a central processing unit and a cache memory connected therewith; a second bus having a dynamic memory access control circuit and an external bus interface connected therewith; a break controller for connecting the first bus and the second bus selectively; a third bus having a peripheral module connected therewith and having a lower-speed bus cycle than the bus cycles of the first and second buses; and a bus state controller for effecting a data transfer and a synchronization between the second bus and the third bus. The single-chip microcomputer has the three divided internal buses to reduce the load capacity upon the signal transmission paths so that the signal transmission can be accomplished at a high speed. Moreover, the peripheral module required to have no operation speed is isolated so that the power dissipation can be reduced.

7 Claims, 44 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 42

---------- KWIC ---------

Detailed Description Text - DETX (27):

For the DMAC, external terminals /DREQ0 and /DREQ1 are input terminals to be fed with a direct memory access (DMA) **transfer request** signal from an external device and correspond to the channel 1 and the channel 2. External terminals /DACK0 and /DACK1 are output terminals for outputting a DMA transfer accept (i.e., DMA transfer acknowledge) signal to the external device having outputted the DMA **transfer request** signal, when the DMA transfer is acknowledged to the DMA **transfer request** signal, and correspond to the channel 1 and the channel 2.

Detailed Description Text - DETX (107):

The bus mode (or transfer mode) is divided into a **cycle steal** mode and a burst mode. In the **cycle steal** mode, after the end of **DMA transfer** of one word, the bus right is released and transferred to another bus master (e.g., CPU). In the burst mode, when a bus right is acquired, the **DMA transfer** is continued till the **transfer** ending condition is satisfied. In case, however, the level of the terminal /DREQ is sampled in the external request mode, the **DMA transfer** is executed according to the level of the terminal /DREQ.

Detailed Description Text - DETX (108):

A **transfer request** is divided into an external request, an internal request from an internal peripheral module, and an auto-request. The external request can start the channel CH0 by the terminal /DREQ0 and the channel CH1 by the terminal /DREQ1. In the sampling of the terminals /DREQ0 and /DREQ1, the fall edge and level can be selected as the select conditions. The internal request from the internal peripheral module is divided into the receive data full of the internal SCI0 and the send data line empty of the internal SCI. These requests are automatically cleared by starting the DMA transfer cycle. For the auto-request, the transfer operation is started by setting the DE bits of the channel control registers DHCRn (0, 1) of the DMAC.

Detailed Description Text - DETX (109):

In case a plurality of channel **transfer requests** are simultaneously made for the DMAC, the transfer channel is determined in accordance with the priority order. This priority order is divided into a priority order fixed mode and an alternate mode. In the priority order fixed mode, the priority order between the individual channels is unchanged. Two channels of 1 or 0 can be fixed to the priority order. In the alternate mode, the priority order is alternated between the channels CH0 and CH1, and the timing for changing the priority order is determined when one transfer unit (byte or word) of the channel CHO or CH1 is ended by the round robin scheduling.

US-PAT-NO: 5640598

DOCUMENT-IDENTIFIER: US 5640598 A

TITLE: Data transfer processing system

DATE-ISSUED: June 17, 1997

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Sato; Fumiki	Hyogo-ken	N/A	N/A	JP
Fujita; Kouichi	Hyogo-ken	N/A	N/A	JP

US-CL-CURRENT: 710/22, 709/212 , 710/1 , 710/48 , 711/100

ABSTRACT:

A transfer processing procedure comprising instruction words is inputted from an external to a program storing area of a DMA processing unit 40 whose functions are arranged to correspond to instructions, and the DMA processing unit 40 performs a transfer process in accordance with the function corresponding to the inputted instruction word.

14 Claims, 15 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 14

---------- KWIC ---------

Brief Summary Text - BSTX (6):

FIG. 15 shows the contents of the control register 5 of the DMA processing unit 4. In FIG. 15, numeral 16 is a bit indicative of the **transfer** unit in the **DMA transfer** where "0" indicates that the **transfer** unit is byte and "1" indicates that the **transfer** unit is word. Further, numeral 17 is a bit indicative of the mode of the **DMA transfer** where "0" indicates the **cycle steal** mode and "1" indicates the burst mode. Here, the **cycle steal** mode is an operational mode for **DMA transfers** to one unit of **transfer** in response to a **DMA transfer request** before returning to the CPU 1, and the burst mode is an operational mode for continuous **transfers** invoked by a **DMA transfer request** without returning to the CPU 1. Moreover, 18a and 18b are indicative of methods of correcting the address of the **transfer** source after the **DMA transfer** is effected in one **transfer** unit where, when "00", the address of the **transfer** source is fixed as it is after the **DMA transfer,** when "01", the address of the **transfer** source is incremented, when "10", the address of the **transfer** source is decremented, and "11" is in an unused state. Further, 19a and 19b show the correcting methods of the address of the transfer destination. When "00", the address of the **transfer** destination is fixed as it is after the **DMA transfer,** when "01", the address of the **transfer** destination is incremented, when "10", the address of the **transfer** destination is decremented, and "11" is in an unused state. Numeral 20 represents a **DMA transfer** effective flag where "0" indicates that the **DMA** processing unit is not used and "1" indicates that it is used.

Brief Summary Text - BSTX (8):

However, when the communication speed is high, there is the possibility that the next data is supplied before reading the received data from the external data communication means 14, and hence the processing procedure due to the interruption can provide an inconvenience. In this case, a processing procedure using the DMA processing unit 4 is required to be effected. Thus, the interruption request signal of the external data communication unit 14 is arranged to become a DMA **transfer request** signal with respect to the DMA processing unit 4. Further, since the initialization of the respective registers is required before use, before performing the above-mentioned procedure, the DMA processing unit 4 writes, through the CPU, the control code "10010000" in the control register 5, the read address of the received data of the external data communication means 14 in the transfer-source address register 6, an adequate address of the memory means 15 in the transfer-destination address register 7, and the number of bytes of the coming data in the transfer count register 8.

Brief Summary Text - BSTX (9):

In response to the reception of the data, the external data communication unit 14 generates an interruption request signal. This signal is inputted as the DMA **transfer request** signal to the DMA processing unit 4. In response to the input, the DMA processing unit 4 supplies a hold request signal to the CPU 1 which in turn interrupts the currently executing process and opens the address bus 11, the data bus 12 and the control signal bus 13 and further supplies a hold acknowledge signal to the DMA processing signal 4. Thus, the DMA processing unit 4 supplies a DMA acknowledge signal to the external data communication unit 14 so that the received signal is read out from the external data communication unit 14 in accordance with the address stored in the transfer-source address register 6 and temporarily stored in the data register 10. Secondly, this data is written in the memory unit 15 in accordance with the address stored in the transfer-destination address register 7. At the same time, the content of the transfer-destination address register 7 is incremented and the content of the transfer count register 8 is decremented. In response to the completion of the DMA transfer in one unit, the DMA processing unit 4 withdraws the hold request signal, and hence the CPU restarts the interrupted process.

Detailed Description Text - DETX (5):

After the execution of "MOV SA.PHI., x", "MOV DA.PHI., y" and "MOV TCQ, Z", the DMA processing unit 40 repeatedly executes the instruction word "WAIT" until inputting a DMA **transfer request** signal to allow the start of the DMA transfer. In response to the reception of data, the external data communication unit 14 generates an interruption request signal. This signal is inputted as the DMA **transfer request** signal to the DMA processing unit 40, whereby the DMA processing unit 40 supplies a hold request signal to the CPU 1 which in turn interrupts the currently executing process and opens the address bus 11, the data bus 12 and the control signal bus 13 and further supplies a hold acknowledge signal to the DMA processing unit 40. In response to the hold acknowledge signal, the DMA processing unit 40 supplies a DMA acknowledge signal to the external data communication means 14 and further executes the next instruction word "LDB AU". In detail, the DMA processing unit 40 executes the function corresponding to this instruction word in the FIG. 2 table. Thus, the received data is read out from the address designated by the transfer-source address register 6 of the external data communication means 14. Secondly, the function in FIG. 2 which corresponds to the instruction word "STB AI" is executed whereby the data read out by the function corresponding to the instruction word "LDB AU" is written at the address designated by the transfer-destination address register 7 of the memory unit 15. At the same

time, the content of the transfer-destination address register 7 is incremented. The content of the transfer count register 8 is decremented by the function corresponding to the final instruction word "LOOP L1, L1", before branching to the label L1, i.e., the initial instruction word "WAIT" stored in the program storing area 22. After the completion of the above-described DMA transfer in one unit, the DMA processing unit 40 withdraws the hold request signal whereby the CPU restarts the interrupted process.

Detailed Description Text - DETX (11):

In response to the reception of data, the external data communication unit 14 generates an interruption request signal. This signal is inputted as the DMA **transfer request** signal to the DMA processing unit 41. The DMA processing unit 41 supplies a hold request signal to the CPU 1. The CPU 1 interrupts the currently executing process and opens the address bus 11, the data bus 12 and the control signal bus 13 and further supplies a hold acknowledge signal to the DMA processing unit 41. In response to the acknowledge signal, the DMA processing signal 41 supplies a DMA acknowledge signal to the external data communication means 14 and further transmits the instruction word "LDB AU" through the instruction word transmitting executing unit 24 to the CPU 1. This instruction word is stored in the instruction register 3 and interpreted by the instruction word interpreting unit 23 on the basis of the tables shown in FIGS. 2 to 4, whereby the received data is read out from the external data communication means 14. Secondly, the instruction word "STB Ai" is transmitted whereby the data read out by the instruction word "LDB AU" is written in the memory unit 15 and at the same time the content of the transfer-destination address register 7 is incremented. The content of the transfer count register 8 is decremented by the final instruction word "LOOP L1, L1", before branching to the label "L1", i.e., the initial instruction word "WAIT". When finding out the instruction word "WAIT", the CPU 1 interrupts the DMA transfer process, thereby completing the DMA transfer in one unit.

Claims Text - CLTX (45):

9. The system of claim 2 wherein the programs for changing contents of data transfer processing include a Wait command as a first instruction word following the initial value setting program for keeping the DMA processing unit to wait for the transfer processing until the DMA processing unit receives a DMA **transfer request** signal as an interrupt processing request signal.

Claims Text - CLTX (48):

12. The system of claim 5 wherein the programs for changing contents of data transfer processing include a Wait command as a first instruction word following the initial value setting program for keeping the DMA processing unit to wait for the transfer processing until the DMA processing unit receives a DMA **transfer request** signal as an interrupt processing request signal.

US-PAT-NO: 5539916

DOCUMENT-IDENTIFIER: US 5539916 A

TITLE: **DMA** control for continuing **transfer** to input/output device in a **cycle steal** mode

DATE-ISSUED: July 23, 1996

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Yamasaki; Takashi	Itami	N/A	N/A	JP
Kuroda; Sachie	Itami	N/A	N/A	JP

US-CL-CURRENT: 710/22, 327/339 , 327/352

ABSTRACT:

A **DMA** control system continuously grants permission to access the I/O device and memory to continue data **transfer in a cycle steal** mode when there is a continuous stream of **DMA** requests from a number of I/O devices by producing a logical sum of the **DMA** requests.

11 Claims, 4 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 4

---------- KWIC ---------

Abstract Text - ABTX (1):

A **DMA** control system continuously grants permission to access the I/O device and memory to continue data **transfer in a cycle steal** mode when there is a continuous stream of **DMA** requests from a number of I/O devices by producing a logical sum of the **DMA** requests.

TITLE - TI (1):

DMA control for continuing **transfer** to input/output device in a **cycle steal** mode

Brief Summary Text - BSTX (8):

If the bus available signal BAK is kept to be L, the **DMA transfer** is carried out in the burst mode. However, since L of the signal BAK is applied to the reset (R) of the request signal generator 10, the signal BAK becomes H at the end of a machine cycle, and the bus request signal BRQ from the request signal generator 10 is inverted to L. Consequently, it cancels the bus available signal BAK and outputs a H signal immediately after one machine cycle which is inherent to the system. Thus, the operation enters a **cycle steal** mode in which one byte of data is transferred in the machine cycle. When a DMA request signal DRQ is outputted from the channel No. 0 of the I/O device after the bus

use permit is transferred to the CPU 1, one byte of data is transferred in the same way as described above.

Brief Summary Text - BSTX (10):

However, the conventional **DMA** controller requires a preparation period t necessary for the initial setting to effect data **transfer** within the period T in which the address output device 3 is given a bus use permit. Since this preparation period t is necessary for each transfer of one byte, the effective data transfer time is (T-t), resulting in the low bus efficiency. Thus, there is a waste of time on the bus for data transfer in the **cycle steal** mode.

Brief Summary Text - BSTX (13):

In accordance with the invention there is provided a direct memory access control system which includes a central processing unit, a random access memory, a plurality of input/output devices, a plurality of buses for interconnecting the central processing unit to the random access memory and the input/output devices, and a direct memory access controller operable in a **cycle steal** mode in which the central processing unit is cut off from the buses while one of the input/output devices is connected to the buses for a machine cycle of the central processing unit (connection conditions). If there is a continuous stream of **DMA** requests the said I/O devices in the **cycle steal** mode, the connections of the buses will be maintained during a period of the continuous stream of **DMA** requests from the I/O devices in the **cycle steal** mode to carry out continuous data **transfer** between the memory and the I/O device.

Brief Summary Text - BSTX (14):

When there is a continuous stream of **DMA** requests during **transfer** of a data unit determined by the machine cycle, the CPU continuously grants permission to the address output device to continuously perform direct data **transfer** between the RAM and the I/O device. When the DMA requests end, the control is returned to an ordinary **cycle steal** mode. In this way, it is possible to eliminate the preparation period which is otherwise necessary before data **transfer in a cycle steal** mode, thus resulting in increased **DMA** data **transfer** efficiency.

Detailed Description Text - DETX (6):

When a DMA **transfer request** signal DRQ (No. 0 request) is outputted from another I/O device, the OR gate 13 outputs the sum signal (No. 0+No. 1) of the request signal DRQ (No. 1) from the I/O device 12a and the request signal DRQ (No. 0) from the I/O device 12b. That is, it continuously outputs L. This signal L, which has little or no influence on the transfer pulse generator 11, is applied to the gate of the tristate buffer 15 so that the tristate buffer is turned off while the sum signal (No. 0+No. 1) is L.

Detailed Description Text - DETX (8):

When there is no DMA **transfer request** DRQ from both channels of the I/O devices 12a and 12b, the output of the OR gate 13 becomes H. This signal H is applied to the gate of the tristate buffer 15 to turn it on. H of the bus available signal BAK from the bus access controller 14 is applied to the reset (R) of the request signal generator 10 so that the bus request signal BRQ from the request signal generator 10 is inverted to L. In this way, the bus access controller 14 quickly cancels the bus available signal BAK and outputs H. Consequently, after a series of DMA transfers, the bus use permit is returned to the CPU.

Detailed Description Text - DETX (10):

Thus, when **DMA** requests are provided continuously from the I/O devices 12, the **cycle steal** mode of a certain period is switched to a continuous **transfer** mode of a various period which is similar to the burst mode, thereby eliminating the preparation period in the data **transfer** initial period.

Detailed Description Text - DETX (11):

As has been described above, according to the invention, when there is a continuous stream of **DMA** requests in a **cycle steal** mode in which the CPU is cut off from the buses while the I/O device is connected to the buses during a machine cycle of the CPU, the bus connection conditions are maintained during the period of the continuous request to carry out continuous data **transfer** between the main memory and the I/O device designated by the address output device so that it is possible to eliminate a waste of time for initial setting which arises in the data **transfer** starting period in the **cycle steal** mode, thereby enhancing the data **transfer** efficiency and the entire function of the information processor.

US-PAT-NO: 5535417

DOCUMENT-IDENTIFIER: US 5535417 A
See image for Certificate of Correction

TITLE: On-chip DMA controller with host computer interface employing boot sequencing and address generation schemes

DATE-ISSUED: July 9, 1996

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Baji; Toru	San Jose	CA	N/A	N/A
Kiuchi; Atsushi	Kunitachi	N/A	N/A	JP

US-CL-CURRENT: 710/22, 710/33 , 710/9

ABSTRACT:

A single chip digital signal processor (DSP) includes memory mapped resources and an on-chip direct memory access controller (DMAC). The memory mapped resources of the DSP include an on-chip program memory, an on-chip data memory, internal registers and memory mapped external memories and peripheral devices. The DMAC includes a host computer interface that processes host originated data transfer commands for transferring data to and from memory mapped resources of the DSP, and commands for setting the mode of operation of the DSP. The DMAC also has a dedicated interrupt controller for handling interrupts from a host computer and from peripheral devices. The DMAC processes interrupts from the host while a primary direct memory access transfer is being performed by the DMAC without having to store address register and count register information in a memory stacking area. As a result, the DMAC can switch from a primary DMA transfer to a host data transfer and back without using any instruction cycles for "overhead" associated with storing and restoring registers in a memory stacking area. The DMAC's host interface is also designed to be connected to a byte-structured boot ROM and the DMAC includes a boot sequencer for automatically loading a boot program from the ROM into the DMAC's on-chip instruction memory whenever the DSP is reset and a boot ROM is connected to the host interface.

12 Claims, 49 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 43

---------- KWIC ---------

Brief Summary Text - BSTX (17):

Accordingly, it is a goal of the present invention to improve DMAC performance and the speed of data transfers. This goal is achieved in part by putting as many functions as possible on-chip, including memories, peripheral interfaces, and the DMA controller. The DMAC runs concurrently with the DSP core and provides for local data transfers between DSP memory mapped resources, including memories and interfaces for various input/output (I/O) peripherals, and accepts host and peripheral data **transfer requests**.

Brief Summary Text - BSTX (18):

To achieve some of these functions, existing DMACs provide a dedicated bus for DMA transfers and respond to interrupts from peripheral I/O devices transferring data. However none of those DMACs interrupt local DMA (e.g., data transfer between memories) to handle peripheral interrupts, nor do they provide dedicated pointers and counters for peripheral interrupts. As a result, peripheral interrupts are handled less than optimally in current on-chip DMACs. Similarly, none of the existing DMACs respond to host interrupts (for data **transfer requests** or commands) in the middle of a local DMA operation, nor do they provide dedicated registers to make host data transfers as efficient as possible. Finally, current on-chip DMACs do not provide a host interface that allows a host to issue DSP commands, or a boot ROM to download programs and data to a stand alone DSP. Failing to integrate such host features in the DMAC means that they must be foregone or provided elsewhere at the cost of additional hardware.

Brief Summary Text - BSTX (19):

It is therefore the object of the present invention to provide an on-chip DMAC that operates concurrently with the DSP core, expedites peripheral and host data **transfer requests,** and provides a host interface with an eye to minimizing on-chip hardware. The specific objects are to provide:

Detailed Description Text - DETX (41):

Referring to FIGS. 1 and 4A-4F the DMAC 3000 controls local data transfers between DSP memory mapped storage resources (including addresses for external peripherals 2300, data memory 1900, instruction memory 1400, and external memory 2500) and between the host 1200 and the DSP Core 3500. The DMAC 3000 handles host and peripheral data **transfer requests** by interrupting local data transfers, provides integrated handling of host commands, and loads programs and data from a byte structured boot ROM into the DSP 1100 when the DSP 1100 is operating as a stand alone processor.

Detailed Description Text - DETX (42):

The DMAC 3000 is primarily occupied with local data transfers. However, the DMAC 3000 can interrupt local transfers to handle data **transfer requests** from up to two on-chip "peripherals" (consisting of serial I/O devices 0 and 1, and the serial communications device) and the host 1200. As will become apparent in the following discussion of the DMAC 3000 functional blocks, the DMAC 3000 handles this diversity of transfers by providing separate registers dedicated to local data transfers, two interrupt driven peripheral transfers, and interrupt-driven host data transfers and commands.

Detailed Description Text - DETX (59):

The typical data transfer mediated by the DMAC is between fast memories, including the on-chip data and instruction memories and the external memory. At various times, the DMAC will also need to handle data **transfer requests** from relatively slower peripherals for external systems 2300 via serial I/O interfaces and a serial communications interface. Referring to FIGS. 4D and 4E, interrupt signals from up to three external peripherals can be separately handled, with each of the peripheral devices having its own Output Enable (for outbound data transfers) and Input Full (for inbound data transfers) interrupts: S0DOE and S0DIF for Serial I/O Peripheral Device 0 2302, S1DOE and S1DIF for Serial I/O Peripheral Device 1 2304, SCDOE and SCDIF for a Serial Communication Device 2306. Serial I/O Peripheral devices 0 and 1 may be CODEC

chips or other devices which access serial digital data. The Serial Communication Device 2306 is typically a device having an RS232 port.

Detailed Description Text - DETX (61):

The DMAC handles these peripheral data **transfer requests** on an interrupt basis using a dedicated DMA interrupt controller 3060. Enabling these transfers is the dedicated peripheral interrupt control register ictr 3022, shown in FIGS. 4B, 4D and 6. Register ictr 3022 provides four fields of information for each of the two possible peripheral interrupts PIRQ0 AND PIRQ1. The DSP core has read/write access to all of these fields of ictr 3022 except the interrupt flags (PIRQ1, PRIQ0) themselves, for which it has only read access.

Detailed Description Text - DETX (85):

(2) fast handling of peripheral data **transfer requests**;

Detailed Description Text - DETX (90):

The most common form of data transfers controlled by the DMAC are local data transfers, which involve the high throughput movement of data between data memory 1900, instruction memory 1400, parallel memory interface 2400 (providing an interface to external memories 2500) and the peripheral devices interface 2200 (providing an interface to external systems 2300). In a typical local data transfer, the DMAC might move a DSP program from a relatively slow external memory 2500 into the faster on-chip instruction memory 1400. Local data transfers, being the most common DMAC data transfers, are interruptable by the less frequent peripheral and host data **transfer requests**.

Detailed Description Text - DETX (108):

There are six interrupts to DMAC operation which are shown in Table 3. These interrupts are one of the unique features of this DSP in that they allow the DSP to quickly interrupt a local data transfer operation to handle host or peripheral data **transfer requests**.

Detailed Description Text - DETX (161):

Host **DMA** is used as a high speed data **transfer** method between host memory 1204 and the DSP. Both data and instructions can be transferred to the DSP using host **DMA transfers**. Therefore, large application programs or data can be down loaded from the host memory, host hard disks, floppy disks or communication networks. Using its own DMA controller 1202 (see FIG. 4D), the host 1200 can access not only DSP on-chip memories and registers, but also DSP external memory 2500. Host **DMA** can be performed in two modes, **cycle steal** mode, where the DSP steals memory access cycles from the host, and burst mode, where the DSP captures the host during the **transfer**.

Detailed Description Text - DETX (163):

Host **DMA transfers** are controlled through the handshake signals, HDREQ, HDACK and HEOD. In the **cycle steal** data transfer, the DMAC requests the next data by setting the HDREQ signal. The host DMA controller then issues the HDACK signal to the DSP acknowledging the request. In the burst transfer mode, HDREQ is held high during the transaction. The falling edge of HDACK initiates the data move within the DSP. When all data **transfer** transactions are

complete, the host **DMA** controller 1202 issues the end of data signal HEOD.

Detailed Description Text - DETX (164):

FIG. 21 shows the DMA procedure in the case of a **cycle steal** DMA read where the host bus is 8-bit. The host first writes the starting address for the **DMA transfer** to the DSP pointer registers. The host also sends the destination or source address of the host memory location and the **DMA transfer** counts to the host DMAC 1202. The host then writes a start command into the host DMAC, after which the DMAC is ready to commence the transfer upon receipt of the HDREQ signal from the DSP.

Detailed Description Text - DETX (166):

In the case of **cycle steal DMA** mode, after each such **transfer** is completed, the DSP sends data **transfer request** signal HDREQ to the host DMAC 1202. Note that where the operation is a DMA write, no data is transferred before HDREQ is sent to the host DMAC.

Detailed Description Text - DETX (170):

Referring to FIG. 22, host Burst **DMA transfers** are similar to the **cycle steal** data **transfers** except that Host burst **transfers** issue only one interrupt to the DSP, either a DOE (signifying a host read) or a DIF (signifying a host write), then **transfer** data continuously until the end of the data is reached. In contrast, **cycle steal** mode **DMA transfers** require interrupting the DSP before every word transferred. Burst mode **transfers** are implemented by the DSP holding the HREQ pin high during the **DMA** transaction and each data move is initiated at the falling edge of the HDACK signal. Further, the DMAC 300 does not decrement an internal count register for host burst **DMA transfers**. Rather, the host computer or the host computer's DMA maintains a counter and sends the DMAC an end of data signal HEOD on the HEOD pin after the last data word has been transferred.

US-PAT-NO: 5513374

DOCUMENT-IDENTIFIER: US 5513374 A

TITLE: On-chip interface and DMA controller with interrupt functions for digital signal processor

DATE-ISSUED: April 30, 1996

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Baji; Toru	San Jose	CA	N/A	N/A

US-CL-CURRENT: 710/26, 710/33 , 710/48

ABSTRACT:

A single chip digital signal processor (DSP) includes memory mapped resources and an on-chip direct memory access controller (DMAC). The memory mapped resources of the DSP include an on-chip program memory, an on-chip data memory, internal registers and memory mapped external memories and peripheral devices. The DMAC includes separate address and count registers for handling a primary data transfer and two interrupt data transfers. The count registers share the same decrementer and the address registers share the same address computation circuit. The DMAC also has a dedicated interrupt controller for handling interrupts from a host computer and from peripheral devices. The DMAC processes interrupts from the host and two peripheral devices while a primary direct memory access transfer is being performed by the DMAC without having to store address register and count register information in a memory stacking area. As a result, the DMAC can switch from a primary DMA transfer to an interrupt DMA transfer or a host DMA transfer and back without using any instruction cycles for "overhead" associated with storing and restoring registers in a memory stacking area. The DMAC also includes a host computer interface that processes host originated data transfer commands for transferring data to and from memory mapped resources of the DSP, and commands for setting the mode of operation of the DSP.

11 Claims, 49 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 43

---------- KWIC ---------

Brief Summary Text - BSTX (17):

Accordingly, it is a goal of the present invention to improve DMAC performance and the speed of data transfers. This goal is achieved in part by putting as many functions as possible on-chip, including memories, peripheral interfaces, and the DMA controller. The DMAC runs concurrently with the DSP core and provides for local data transfers between DSP memory mapped resources, including memories and interfaces for various input/output (I/O) peripherals, and accepts host and peripheral data **transfer requests**.

Brief Summary Text - BSTX (18):

To achieve some of these functions, existing DMACs provide a dedicated bus for DMA transfers and respond to interrupts from peripheral I/O devices transferring data. However none of those DMACs interrupt local DMA (e.g., data transfer between memories) to handle peripheral interrupts, nor do they provide dedicated pointers and counters for peripheral interrupts. As a result, peripheral interrupts are handled less than optimally in current on-chip DMACs. Similarly, none of the existing DMACs respond to host interrupts (for data **transfer requests** or commands) in the middle of a local DMA operation, nor do they provide dedicated registers to make host data transfers as efficient as possible. Finally, current on-chip DMACs do not provide a host interface that allows a host to issue DSP commands, or a boot ROM to download programs and data to a stand alone DSP. Failing to integrate such host features in the DMAC means that they must be foregone or provided elsewhere at the cost of additional hardware.

Brief Summary Text - BSTX (19):

It is therefore the object of the present invention to provide an on-chip DMAC that operates concurrently with the DSP core, expedites peripheral and host data **transfer requests,** and provides a host interface with an eye to minimizing on-chip hardware. The specific objects are to provide:

Detailed Description Text - DETX (41):

Referring to FIGS. 1 and 4A-4F the DMAC 3000 controls local data transfers between DSP memory mapped storage resources (including addresses for external peripherals 2300, data memory 1900, instruction memory 1400, and external memory 2500) and between the host 1200 and the DSP Core 3500. The DMAC 3000 handles host and peripheral data **transfer requests** by interrupting local data transfers, provides integrated handling of host commands, and loads programs and data from a byte structured boot ROM into the DSP 1100 when the DSP 1100 is operating as a stand alone processor.

Detailed Description Text - DETX (42):

The DMAC 3000 is primarily occupied with local data transfers. However, the DMAC 3000 can interrupt local transfers to handle data **transfer requests** from up to two on-chip "peripherals" (consisting of serial I/O devices 0 and 1, and the serial communications device) and the host 1200. As will become apparent in the following discussion of the DMAC 3000 functional blocks, the DMAC 3000 handles this diversity of transfers by providing separate registers dedicated to local data transfers, two interrupt driven peripheral transfers, and interrupt-driven host data transfers and commands.

Detailed Description Text - DETX (59):

The typical data transfer mediated by the DMAC is between fast memories, including the on-chip data and instruction memories and the external memory. At various times, the DMAC will also need to handle data **transfer requests** from relatively slower peripherals for external systems 2300 via serial I/O interfaces and a serial communications interface. Referring to FIGS. 4D and 4E, interrupt signals from up to three external peripherals can be separately handled, with each of the peripheral devices having its own Output Enable (for outbound data transfers) and Input Full (for inbound data transfers) interrupts: S0DOE and S0DIF for Serial I/O Peripheral Device 0 2302, S1DOE and S1DIF for Serial I/O Peripheral Device 1 2304, SCDOE and SCDIF for a Serial Communication Device 2306. Serial I/O Peripheral devices 0 and 1 may be CODEC chips or other devices which access serial digital data. The Serial Communication Device 2306 is typically a device having an RS232 port.

Detailed Description Text - DETX (61):

The DMAC handles these peripheral data **transfer requests** on an interrupt basis using a dedicated DMA interrupt controller 3060. Enabling these transfers is the dedicated peripheral interrupt control register ictr 3022, shown in FIGS. 4B, 4D and 6. Register ictr 3022 provides four fields of information for each of the two possible peripheral interrupts PIRQ0 AND PIRQ1. The DSP core has read/write access to all of these fields of ictr 3022 except the interrupt flags (PIRQ1, PRIQ0) themselves, for which it has only read access.

Detailed Description Text - DETX (85):

(2) fast handling of peripheral data **transfer requests**;

Detailed Description Text - DETX (90):

The most common form of data transfers controlled by the DMAC are local data transfers, which involve the high throughput movement of data between data memory 1900, instruction memory 1400, parallel memory interface 2400 (providing an interface to external memories 2500) and the peripheral devices interface 2200 (providing an interface to external systems 2300). In a typical local data transfer, the DMAC might move a DSP program from a relatively slow external memory 2500 into the faster on-chip instruction memory 1400. Local data transfers, being the most common DMAC data transfers, are interruptable by the less frequent peripheral and host data **transfer requests**.

Detailed Description Text - DETX (108):

There are six interrupts to DMAC operation which are shown in Table 3. These interrupts are one of the unique features of this DSP in that they allow the DSP to quickly interrupt a local data transfer operation to handle host or peripheral data **transfer requests**.

Detailed Description Text - DETX (159):

Host **DMA** is used as a high speed data **transfer** method between host memory 1204 and the DSP. Both data and instructions can be transferred to the DSP using host **DMA transfers**. Therefore, large application programs or data can be down loaded from the host memory, host hard disks, floppy disks or communication networks. Using its own DMA controller 1202 (see FIG. 4D), the host 1200 can access not only DSP on-chip memories and registers, but also DSP external memory 2500. Host **DMA** can be performed in two modes, **cycle steal** mode, where the DSP steals memory access cycles from the host, and burst mode, where the DSP captures the host during the **transfer**. FIG. 20 shows a host computer with its own DMA controller connected to the DSP of the present invention. The Host DMAC 1202 can either reside within the host 1200 or be connected outside the host 1200. The DSP's host data bus HD0-7 (when host bus width signal HBW=0) or HD0-15 (when HBW=1) is connected to a corresponding portion of the host's own data bus, either bits 0-7 or bits 0-15. Rather than issuing addresses to the DMAC via the address bus, the host issues addresses to the DSP via the host pointer register hp 3024. During a host **DMA transfer,** the host increments the value in hp 3024 after every three **transfers** (if HBW=0) or every two **transfers** (if HBW=1), which corresponds to the number of **transfers** required to write to the 24-bit DSP registers.

Detailed Description Text - DETX (160):

Host **DMA transfers** are controlled through the handshake signals, HDREQ, HDACK and HEOD. In the **cycle steal** data transfer, the DMAC requests the next data by setting the HDREQ signal. The host DMA controller then issues the HDACK signal to the DSP acknowledging the request. In the burst transfer mode, HDREQ is held high during the transaction. The falling edge of HDACK initiates the data move within the DSP. When all data **transfer** transactions are complete, the host **DMA** controller 1202 issues the end of data signal HEOD.

Detailed Description Text - DETX (161):

FIG. 21 shows the DMA procedure in the case of a **cycle steal** DMA read where the host bus is 8-bit. The host first writes the starting address for the **DMA transfer** to the DSP pointer registers. The host also sends the destination or source address of the host memory location and the **DMA transfer** counts to the host DMAC 1202. The host then writes a start command into the host DMAC, after which the DMAC is ready to commence the transfer upon receipt of the HDREQ signal from the DSP.

Detailed Description Text - DETX (163):

In the case of **cycle steal DMA** mode, after each such **transfer** is completed, the DSP sends data **transfer request** signal HDREQ to the host DMAC 1202. Note that where the operation is a DMA write, no data is transferred before HDREQ is sent to the host DMAC.

Detailed Description Text - DETX (167):

Referring to FIG. 22, host Burst **DMA transfers** are similar to the **cycle steal** data **transfers** except that Host burst **transfers** issue only one interrupt to the DSP, either a DOE (signifying a host read) or a DIF (signifying a host write), then **transfer** data continuously until the end of the data is reached. In contrast, **cycle steal** mode **DMA transfers** require interrupting the DSP before every word transferred. Burst mode **transfers** are implemented by the DSP holding the HREQ pin high dudng the **DMA** transaction and each data move is initiated at the falling edge of the HDACK signal. Further, the DMAC 300 does not decrement an internal count register for host burst **DMA transfers**. Rather, the host computer or the host computers DMA maintains a counter and sends the DMAC an end of data signal HEOD on the HEOD pin after the last data word has been transferred.

PGPUB-DOCUMENT-NUMBER: 20020133645

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20020133645 A1

TITLE: Direct memory access controller for converting a transfer mode flexibly in accordance with a data transfer counter value

PUBLICATION-DATE: September 19, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY
RULE-47			
Ku, Yong-Guen	Kyungki-do		KR

US-CL-CURRENT: 710/22

ABSTRACT:

A DMA controller includes a burst/single mode control circuit for automatically converting a DMA transfer operation mode to a burst mode and/or a single mode regardless of a data transfer counter value, and for performing the DMA transfer operation. The burst/single mode control circuit carries out the burst mode DMA transfer operation without the need for a control operation of the CPU a number of times corresponding to a quotient which is the result that the data transfer counter value divided by the burst length, and carries out successively the single mode DMA transfer operation by the number of times corresponding to the remainder of the division.

---------- KWIC ---------

Summary of Invention Paragraph - BSTX (7):

[0005] As well known to this art, **DMA transfer** generally includes two modes of **DMA transfer,** single mode and **burst mode**. If a precedence **interrupt** is generated, the single mode first performs the **interrupt** operation, and then transmits data. In contrast, the **burst mode** transmits data completely without being interrupted during the transfer operation of an entire data block. **Transfer** capacity of the system bus of the **DMA** controller thus depends on the application of the **burst mode**. Contemporary DMA controllers generally apply the **burst mode,** and are operable with various sizes of transmitted data (e.g. byte, half-word, and a word) and various burst lengths (e.g. 4, 8, and 16 length).

Detail Description Paragraph - DETX (7):

[0021] For example, a **burst mode** operation of the conventional **DMA** controller is only carried out in the case where the data **transfer** counter value is evenly divisible by the burst length. However, assuming the data **transfer** counter value is not evenly divisible by the burst length, the **DMA** controller 100 of the present invention performs the **burst mode** operation to the extent that this is possible, and then transmits any remaining data according to the single mode. These mode converting processes are automatically performed in the **transfer** mode automatic converting unit 32 in response to a **transfer** counter value TCRV of data stored in the third and

fourth registers 43 and 44 and control signal DCRV for **DMA transfer** operation. After the **DMA transfer** operation is completed, an **interrupt** signal TREI for informing the complete **DMA transfer** operation is provided to the CPU (not shown) through the control unit 31. The detailed burst/single mode control circuit 30 is as follows.

Detail Description Paragraph - DETX (15):

[0029] The decrementor 33, by the selected **transfer** operation mode flag signal STMf from the mode converter 35 and control signal of the control unit 31, decreases a predetermined value (e.g. burst length in the **burst mode** or `1` in the single mode) from the data **transfer** counter value TCRV in accordance with the current **DMA transfer** operation mode. When the TCRV decreased by the decrementor 33 becomes `0`, the control unit 31 provides an **interrupt** signal TEI informing an end of the **DMA transfer** operation to the CPU (not shown) via the system bus interface 10.

US-PAT-NO: 6070210

DOCUMENT-IDENTIFIER: US 6070210 A

TITLE: Timing mode selection apparatus for handling both burst mode data and single mode data in a DMA transmission system

DATE-ISSUED: May 30, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Cheon; Mu-Cheol	Gumi	N/A	N/A	KR

US-CL-CURRENT: 710/105, 710/22

ABSTRACT:

A timing mode selection apparatus for use in a DMA transmission system having a DMA device and an input/output device directly accessible to a memory. The timing mode selection apparatus includes a microprocessor for generating a mode selection control signal for switching a timing mode. A first buffer transfers a single mode data acknowledgment signal from the DMA device to the input/output device in response to the mode selection control signal having a first logic state. A second buffer transfers a burst mode data acknowledgment signal from the DMA device to the input/output device in response to the mode selection control signal having a second logic state. An inverter inverts the mode selection control signal output from the microprocessor. The inverted mode selection control signal is applied to a control terminal of the second buffer. In this manner, the mode selecting apparatus may freely switch the DMA transmission system between a burst mode and a single mode.

8 Claims, 2 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 2

---------- KWIC ---------

Brief Summary Text - BSTX (7):

The following patents are considered to be representative of the prior art relative to the present invention, and are burdened by the disadvantage discussed above: U.S. Pat. No. 5,701,516 to Cheng et al., entitled High-Performance Non-Volatile RAM Protected Write Cahce Accelerator System Employing DMA And Data Transferring Scheme, U.S. Pat. No. 5,669,014 to Iyengar et al., entitled System And Method Having Processor With Selectable Burst Or No-Burst Write Back Mode Depending Upon Signal Indicating The System Is configured To Accept Bit Width Larger Than The Bus Width, U.S. Pat. No. 5,696,917 to Mills etal., entitled Method And Apparatus For Performing Burst Read Operations In An Asynchronous Nonvolatile Memory, U.S. Pat. No. 5,642,386 to Rocco Jr., entitled Data Sampling Circuit For A **Burst Mode** Communication System, U.S. Pat. No. 5,634,139 to Takita, entitled Microprocessor Using Feedback Memory Address To Internally Generate Bust Mode Transfer Period Signal For Controlling **Burst Mode** Data Transfer To External Memory, U.S. Pat. No. 5,634,099 to Andrews et al., entitled Direct Memory

Access Unit For Transferring Data Between Processor Memories In Multiprocessing Systems, U.S. Pat. No. 5,613,162 to Kabenjian, entitled Method And Apparatus For Performing Efficient Direct Memory Access Data Transfers, U.S. Pat. No. 5,590,286 to Mehring et al., entitled Method And Apparatus For The Pipelining OfData During Direct Memory Accesses, U.S. Pat. No. 5,559,990 to Cheng et aL, entitled Memories With **Burst Mode** Access, U.S. Pat. No. 5,513,374 to Baji, entitled On-Chip Interface And DMA Controller With **Interrupt** Functions For Digital Signal Processor, U.S. Pat. No. 5,453,957 to Norris et al., entitled Memory Architecture For **Burst Mode** Access, U.S. Pat. No. 5,410,656 to King et al., entitled Work Station Interfacing Means Having **Burst Mode** Capability, U.S. Pat. No. 5,347,643 to Kondo et al., entitled Bus System For Coordinating Internal And External Direct Memory Access Controllers, U.S. Pat. No. 5,287,486 to Yamasaki etal., entitled **DMA** Controller Using A Programmable Timer, A **Transfer** Counter and An Or Logic Gate To Control Data **Transfer** Interrupts, U.S. Pat. No. 5,175,825 to Sarr, entitled High Speed, Flexible Source/Destination Data Burst Direct Memory Access Controller, U.S. Pat. No. 4,999,769 to Costers et al., entitled System With Plural Clocks For Bidirectional Information Exchange Between DMA Controller And I/O Devices Via DMA Bus, U.S. Pat. No. 4,799,199 to Scales III et al., entitled Bus Master Having Burst Transfer Mode, U.S. Pat. No. 4,530,053 to Kriz et al., entitled **DMA** Multimode **Transfer** Controls, U.S. Pat. No. 4,403,282 to Holberger et al., entitled Data Processing System Using A High Speed Data Channel For Providing Direct Memory Access For Block Data Transfers, and U.S. Pat. No. 4,084,154 to Panigrahi, entitled Charge Coupled Device Memory System With **Burst Mode**.

US-PAT-NO: 6006303

DOCUMENT-IDENTIFIER: US 6006303 A

TITLE: Priority encoding and decoding for memory architecture

DATE-ISSUED: December 21, 1999

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Barnaby; Michael J.	Marlborough	MA	N/A	N/A
Mammen; Abe	Marlborough	MA	N/A	N/A

US-CL-CURRENT: 710/244, 710/240 , 710/40 , 710/41 , 711/151

ABSTRACT:

A shared resource access priority encoding/decoding and arbitration scheme takes into account varying device requirements, including latency, bandwidth and throughput. These requirements are stored and are dynamically updated based on changing access demand conditions.

7 Claims, 5 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 4

---------- KWIC ---------

Brief Summary Text - BSTX (8):

Another U.S. Pat. No. 5,438,666 "SHARED MEMORY BUS SYSTEM FOR ARBITRATING ACCESS CONTROL AMONG CONTENDING MEMORY REFRESH CIRCUITS, PERIPHERAL CONTROLLERS, AND BUS MASTERS" relates to an arbitration system for a shared address, data and control bus that provides **burst mode** operations for transferring data between a peripheral device and memory via a bus master. The arbitration system is responsive to high priority bus activities, such as memory refresh cycles and **DMA** cycles to temporarily **transfer** control of the shared bus from the bus master to a circuit controlling the high priority activity. The arbitration system further includes timing circuits to assure that a bus master transferring data in the **burst mode** does not retain control of the shared bus for an excessive amount of time. The patent is directed to overcoming disadvantages of a "cycle stealing" system where the memory refresh operation uses a shared bus, in particular, excessive time overhead, **burst mode transfer** limitations in performing **DMA**. The object is to provide faster and more efficient access to the shared address, data and control bus while maintaining the operational integrity of the microprocessor based computer system. The bus arbitration circuit of the system includes logic for protecting the integrity of the shared bus so as to prevent a bus master from obtaining access to the shared bus and retaining control of the shared bus to the exclusion of the microprocessor. As one form of protection, the bus arbitration circuit monitors **interrupt** requests to the microprocessor and grants control of the shared bus to the microprocessor so that the microprocessor can service the **interrupt** requests.

US-PAT-NO: 5287486

DOCUMENT-IDENTIFIER: US 5287486 A

TITLE: DMA controller using a programmable timer, a transfer counter and an or logic gate to control data transfer interrupts

DATE-ISSUED: February 15, 1994

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Yamasaki; Takashi	Itami	N/A	N/A	JP
Kuroda; Sachie	Itami	N/A	N/A	JP

US-CL-CURRENT: 710/22

ABSTRACT:

A DMA controller interrupts data transfer as needed to transfer the bus use permit to the CPU and resumes data transfer when the CPU completes the memory use in the burst mode in which the predetermined number of words is transferred between the I/O device and the memory.

1 Claims, 4 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 4

---------- KWIC ---------

Brief Summary Text - BSTX (7):

The operation will be described with reference to FIG. 4. First of all, one of the I/O devices generates a **DMA** request signal DRQ (No. 1) for data **transfer** with the RAM 2. This is a negative logic signal and is represented by $\overline{DRQ}$. When the DMA request signal DRQ is applied to the request signal generator 10, the request signal generator 10 becomes H and stable there and outputs a bus request signal BRQ to the bus access controller 11. If there is neither DRAM refresh request r with high **interrupt** priority nor external HOLD request, the bus access controller 11 sends bus available signals BAK-A and BAK-B to the DMA controller 17 and the CPU 1, respectively. The CPU 1 then cuts off the data bus 4, the address bus 5, and the control bus 6 to stop the use of data from the RAM 2 The DMA controller 17, on the other hand, outputs to the address output device 3 an acknowledge signal DAK indicating that the buses 4-6 are available. The requesting I/O device 12 identified by the address output device 3 then enters a **burst mode** for a certain period in which 255 bytes of data are transferred directly to the RAM 2.

US-PAT-NO: 4953103

DOCUMENT-IDENTIFIER: US 4953103 A

TITLE: Page printer

DATE-ISSUED: August 28, 1990

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Suzuki; Masahiro	Tokyo	N/A	N/A	JP

US-CL-CURRENT: 358/1.16, 358/1.14 , 358/1.15

ABSTRACT:

A page printer with an internal CPU, **interrupt** controller, and DMAC has a **DMA** stopping circuit in the DMAC, the function of which is to stop a **DMA transfer** in response to a **DMA** stop signal and restart it in response to a **DMA** restart signal. The DMA signal is generated by the **interrupt** controller when it receives certain **interrupt** requests, such as communication **interrupt** requests, enabling the CPU to service these requests promptly. The DMA restart signal is generated at the end of the **interrupt** service routine. This arrangement permits **DMA transfers** to be performed in **burst mode,** stopping only when urgent **interrupt** service is required. **Burst-mode** DMA improves the speed of operation of the printer.

8 Claims, 6 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 5

---------- KWIC ---------

Abstract Text - ABTX (1):

A page printer with an internal CPU, **interrupt** controller, and DMAC has a **DMA** stopping circuit in the DMAC, the function of which is to stop a **DMA transfer** in response to a **DMA** stop signal and restart it in response to a **DMA** restart signal. The DMA signal is generated by the **interrupt** controller when it receives certain **interrupt** requests, such as communication **interrupt** requests, enabling the CPU to service these requests promptly. The DMA restart signal is generated at the end of the **interrupt** service routine. This arrangement permits **DMA transfers** to be performed in **burst mode,** stopping only when urgent **interrupt** service is required. **Burst-mode** DMA improves the speed of operation of the printer.

Detailed Description Text - DETX (32):

A page printer employing the multi-channel DMAC circuit in FIG. 6 efficiently performs not only bit block **transfers** from the character pattern memory to the frame buffer memory but a variety of other **DMA transfers** as well, performing all these **DMA transfers in burst mode,** stopping them only when necessary to allow the CPU to service an **interrupt** request.